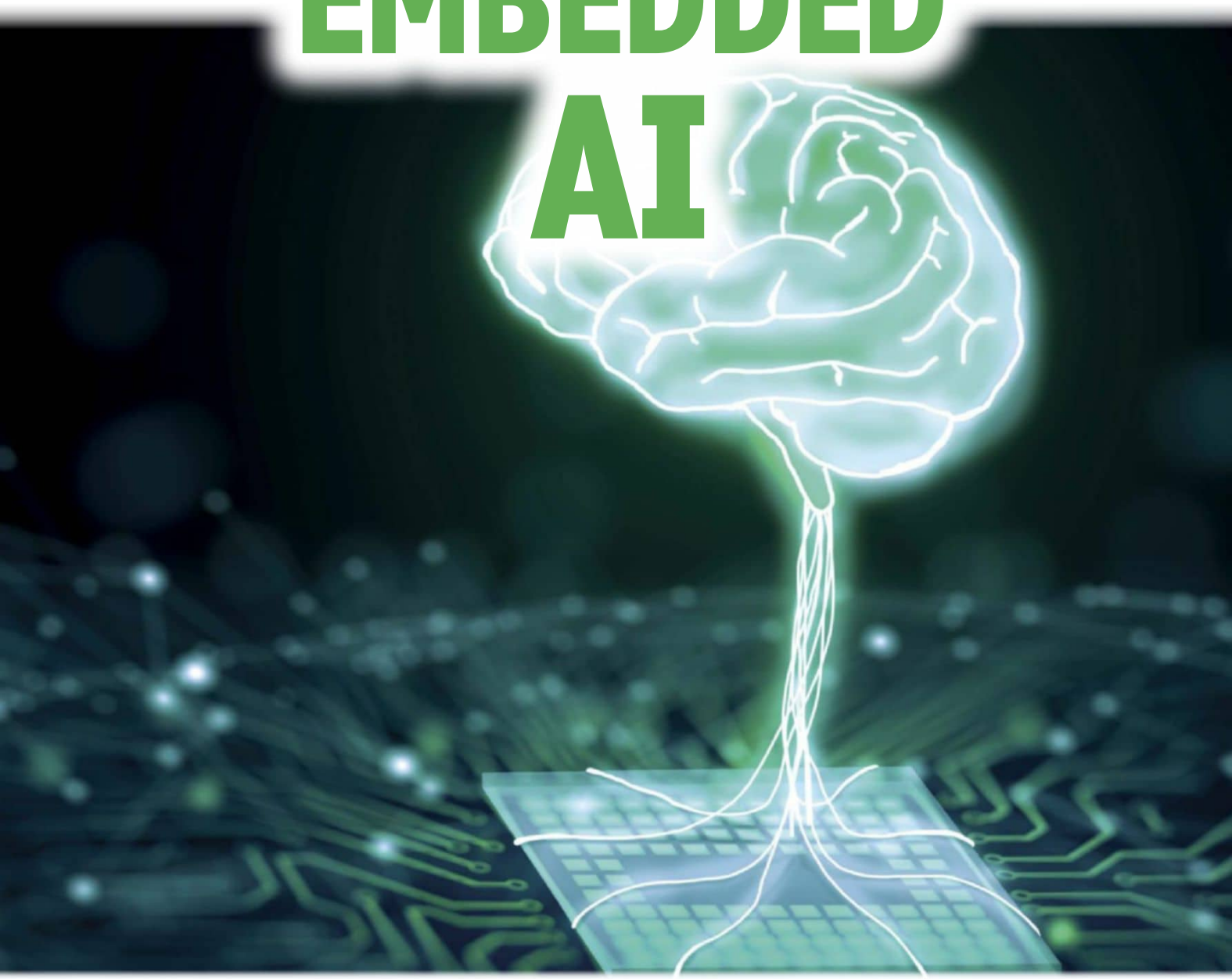
circuitcellar.com

EMBEDDED AI
OCTOBER 2025
ISSUE 423

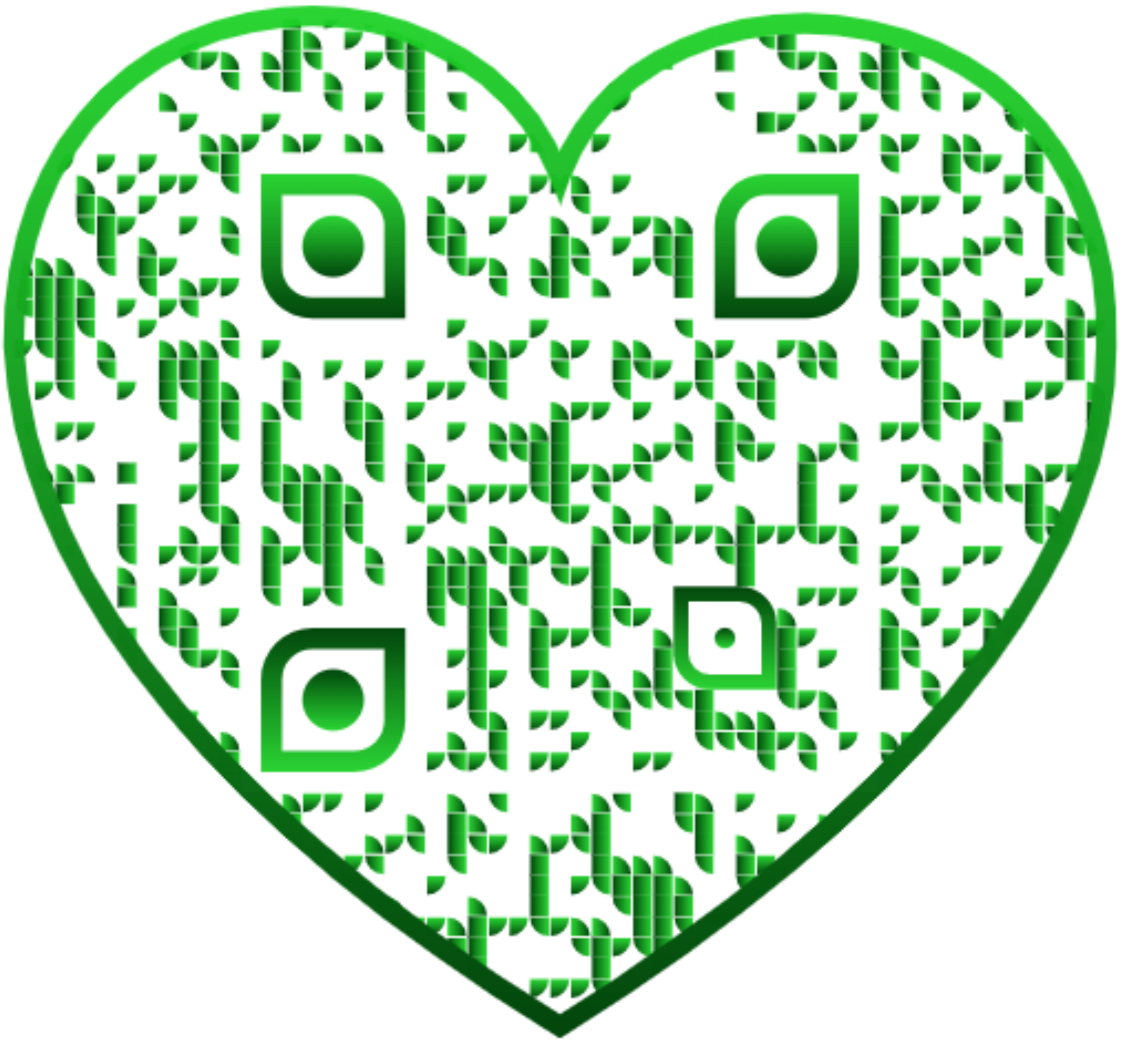
circuit cellar

Inspiring the Evolution of Embedded Design

EMBEDDED AI



MEMS Sensors | Encryption Keeping Your Pi Safe | Pi Retro Game Project | Reliable Communication Mesh Networks | Migrating Legacy Protocols to MQTT | Automating Metrics in Embedded Development | Choosing an Audio Amplifier | Binary Fixed-Point Arithmetic | The Future of Microsegmentation Cyber Defense



Imagine what you will

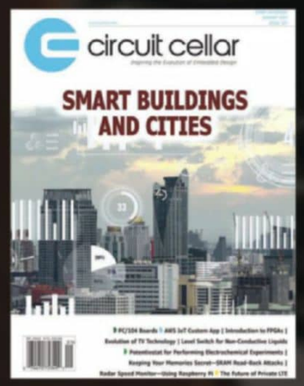
Discover

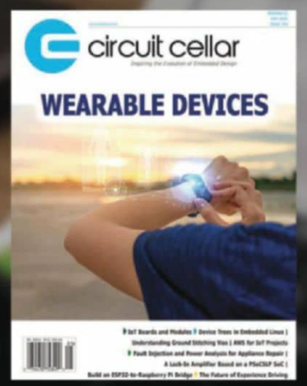
400+ ISSUES

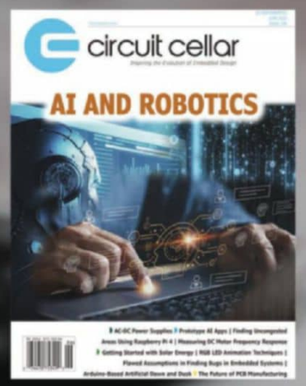
CC VAULT

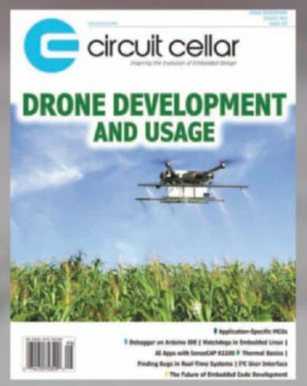
Get the complete *Circuit Cellar* issue archive and article code stored on a stylish, durable and portable USB flash drive. You can easily keep your CC Vault archive up to date by purchasing subsequent issues from our webshop or by downloading issues with a *Circuit Cellar* Digital Subscription. Issues appear in searchable PDF format.

cc-webshop.com









Complete archive includes PDFs of all issues in print through date of purchase.

INPUT Voltage

AI Coding Has Limits

Are you feeling the vibe? I'm not talking about the atmosphere down at the restaurant that's too busy to take your reservation—I'm talking about the vibe that seems like it's putting programmers out of business faster than a return to the abacus. Vibe Programming is a real thing; whether it's a really good thing remains a very active question.

Promises, Promises: Vibe coding is the current term for using a well-crafted prompt to get an large language model (LLM) AI engine to write software for you. It seems to finally fulfill the software promise launched with COBOL: Simply tell the computer what you want it to do in plain language and your algorithmic wish will be granted. No more learning complicated programming languages, no more late nights counting semi-colons, no more taking the time to set up the scripts to link, load, and compile code—just plain, ordinary words typed into a prompt screen.

It's a vision that many employers love. They love it so much, in fact, that in some places it's upending the job market for entry-level programmers with fresh computer science degrees. I know developers at a Very Large Software Company™ who use AI help in their daily work, getting the AI engine's suggestion of code to solve a particular problem. They're highly skilled professionals who are working to improve their productivity, not managers trying to avoid hiring newly minted coders. I think the seasoned folks may be onto something.

AI (of any sort) can be seen as one of two broad things: An assistant for humans or a replacement for humans. In the enterprise IT world, early returns indicate that the current generation of AI is reasonably useful as the first, and an over-hyped disappointment as the second.

Catching the Vibe: So, how should embedded systems programmers be using AI? In some of the tests I've run, I've found that it tends to deliver acceptable results for very simple tasks. For more complex programs, it delivers beautifully structured code that might or might not actually work. In a number of well-known examples, AI has tended to make up libraries to solve problems rather than using real code. While I'll admit that I would probably be a more productive coder if I could just make up a library when I need a function, it's a reminder that humans still have to check AI's work.

As an assistant, though, AI isn't bad. If I think of it as an enhanced version of the IDE that gives me a framework in which to write code, AI helps me be more productive, getting me started faster than I tend to be when I'm just looking at a blank editing screen. I expect AI to get better and better at this kind of assistance, providing more complex code fragments that I can put together to make complete solutions.

The danger to our industry is that cost-cutting administrators will decide that AI is "good enough" for tasks that are beyond its capabilities, creating difficult quality control tasks for developers while cutting off the path for new programmers to enter the workforce. AI can be a great tool—that's how it should be seen.

This Month's Questions: So, are you using AI in your embedded software development? Do you see it as a tool that's part of an IDE or as a fellow developer delivering complete code? Oh, and one other thing: Do you use AI in other aspects of your work? I'd like to know how you see LLM AI in general.

Your answers to my earlier questions have been great—please keep them coming. And let me know how we're doing here at *Circuit Cellar*; we have a lot of great things planned for the future and would love to have your input be part of them.

Curtis Franklin
Editor-In-Chief
cfranklin@circuitcellar.com
LinkedIn: https://www.linkedin.com/in/curtisfranklin/
Facebook: https://www.facebook.com/curtis.f.franklin
Instagram: https://www.instagram.com/curt_franklin/
Mastodon: https://mastodon.sdf.org/@Kg4gwa
Bluesky: https://bsky.app/kg4gwa.bsky.social

Issue 423 October 2025 | ISSN 1528-0608

CIRCUIT CELLAR® (ISSN 1528-0608) is published monthly by:

KCK Media Corp.
PO Box 417, Chase City, VA 23924

Periodical rates paid at Chase City, VA, and additional offices. One-year (12 issues) subscription rate US and possessions $50, Canada $65, Foreign/ ROW $75. All subscription orders payable in US funds only via Visa, MasterCard, international postal money order, or check drawn on US bank.

SUBSCRIPTION MANAGEMENT

Online Account Management: circuitcellar.com/account
Renew | Change Address/E-mail | Check Status

CUSTOMER SERVICE

E-mail: customerservice@circuitcellar.com
Phone: 434.533.0246
Mail: Circuit Cellar, PO Box 417, Chase City, VA 23924

Postmaster: Send address changes to
Circuit Cellar, PO Box 417, Chase City, VA 23924

NEW SUBSCRIPTIONS

circuitcellar.com/subscription

ADVERTISING

Contact: Hugh Heinsohn
Phone: 757-525-3677
Fax: 888-980-1303
E-mail: hheinsohn@circuitcellar.com
Advertising rates and terms available on request.

NEW PRODUCTS

Contact: Kirsten Campbell
E-mail: product-editor@circuitcellar.com

HEAD OFFICE

KCK Media Corp.
PO Box 417
Chase City, VA 23924
Phone: 434-533-0246

COPYRIGHT NOTICE

Entire contents copyright © 2025 by KCK Media Corp. All rights reserved. Circuit Cellar is a registered trademark of KCK Media Corp. Reproduction of this publication in whole or in part without written consent from KCK Media Corp. is prohibited.

DISCLAIMER

KCK Media Corp. makes no warranties and assumes no responsibility or liability of any kind for errors in these programs or schematics or for the consequences of any such errors printed in Circuit Cellar®. Furthermore, because of possible variation in the quality and condition of materials and workmanship of reader-assembled projects, KCK Media Corp. disclaims any responsibility for the safe and proper function of reader-assembled projects based upon or from plans, descriptions, or information published in Circuit Cellar®.

The information provided in Circuit Cellar® by KCK Media Corp. is for educational purposes. KCK Media Corp. makes no claims or warrants that readers have a right to build things based upon these ideas under patent or other relevant intellectual property law in their jurisdiction, or that readers have a right to construct or operate any of the devices described herein under the relevant patent or other intellectual property law of the reader's jurisdiction. The reader assumes any risk of infringement liability for constructing or operating such devices.

© KCK Media Corp. 2025 Printed in the United States

OUR NETWORK

circuit cellar · LinuxGizmos · audioXpress · VOICE COIL · LIS LOUDSPEAKER INDUSTRY SOURCEBOOK

SUPPORTING COMPANIES

Beningo Embedded Group	29
BusBoard Prototype Systems	23
e-con Systems	62
EmbeddedTS	61, C4
Extreme Engineering Solutions, Inc.	49
Front Panel Express	21
Saelig	43
SeaFire Micros, Inc.	C3
Siborg Systems, Inc.	17
STMicroelectronics	45
Texas Instruments	11

NOT A SUPPORTING COMPANY YET?
Contact Hugh Heinsohn
hugh@circuitcellar.com,
Phone: 757-525-3677,
Fax: 888-980-1303
to reserve space in the
next issue of *Circuit Cellar.*

THE TEAM

FOUNDER	Steve Ciarcia
PUBLISHER	KC Prescott
CONTROLLER	Chuck Fellows
EDITOR-IN-CHIEF	Curtis Franklin
CONTRIBUTING EDITOR	Brian Millier
CONTENT MANAGER- EU	Jurij Mikeln
SENIOR ASSOCIATE EDITOR	Shannon Becker
TECHNICAL COPY EDITOR	Carol Bower
PROJECT EDITORS	Ken Davidson
	David Tweed
MARKETING MANAGER	Tori Zienka
ADVERTISING COORDINATOR	Diana Winsor
ADVERTISING SALES REP.	Hugh Heinsohn

COLUMNISTS
Stuart Ball (Start to Finish)
Jacob Beningo (Modern Embedded Insights)
Bob Japenga (Embedded in Thin Slices)
Brian Millier (Picking Up Mixed Signals)
Colin O'Flynn (Embedded Systems Essentials)

FEATURES

4 **Keeping Your Pi Safe**
Part 1—Basic Encryption Functions with the TPM9670 Chip
By Gamal Labib

12 **Synth Hero: Where Synth Legends Are Made**
Creating a Retro Game Using a Raspberry Pi RP2040 MCU
By Chukwuemeka Adebi and Sergio Diaz

18 **Mesh Networks**
Reliable, Decentralized Communication Where Single Routers Fail
By Brian Chee

24 **MQTT in the Real World**
Migrating Legacy Protocols to MQTT
By Ignacio Siccardi

30 TECHNOLOGY FEATURE
Embedded AI
AI Comes to the Operational Network's Edge
By Curtis Franklin

36 DATASHEET
MEMS Sensors
MEMS Chips Bring Programs and Action Together
By Curtis Franklin

COLUMNS

40 Modern Embedded Insights
Automating Metrics in Modern Embedded Development
By Jacob Beningo

46 Start to Finish
Audio Power Amplifiers
Choosing an Audio Amplifier for Your Design
By Stuart Ball

52 Embedded in Thin Slices
Binary Fixed-Point Arithmetic
Extracting Signals from Noise and Other Distractions
By Bob Japenga

63 TECH THE FUTURE
The Future of Microsegmentation Cyber Defense
What Is OT Breach-Ready Cyber Defense?
Microsegmentation Is Strategic Defense Against Intrusion Damage
By Agnidipta Sarkar

56 : PRODUCT NEWS
62 : TEST YOUR EQ

@editor_cc @circuitcellar · circuitcellar · circuitcellar

FEATURES

Keeping Your Pi Safe

Part 1—Basic Encryption Functions with the TPM9670 Chip

By Gamal Labib

The Raspberry Pi has a wide range of IoT applications in personal and industrial domains. Remote access to such devices makes them vulnerable to security risks that can affect their credibility and operation. Meeting international security standards such as Common Criteria, FIPS, ISO, and IEC requires hardening the software and the hardware of such devices to eliminate potential risks. In Part 1 of a two-part article series, Gamal shares his experience applying the Infineon-based TPM9670 encryption module to make Pi security easier and more effective.

Security processors are regular components of modern computers. Modern laptops include a processor called a "Trusted Platform Module" or "TPM" that adds a security layer based on encryption. This TPM can encrypt data on local storage and while it is being sent and received via the Internet. It also ensures secure booting of the computer against loading malicious software. Fortunately, TPMs are not confined to end-user computing facilities, and they can be found in standalone versions suitable for single-board computers (SBCs).

I found that an Infineon TPM9670 breakout board can add significant security to a Raspberry Pi 4. I deployed the board with the help of manufacturer documentation, Raspberry Pi Forums, and ChatGPT.

In the first article on this security solution, I present the basic encryption functions that can be executed in a command shell for the Embedded Linux TPM Toolbox 2 (ELTT2). I also demonstrate the TPM Software Stack (TSS) capabilities that may be embedded in user applications for signing and verifying data, and file and disk encryption.

In the second article, I will discuss the role of TPM in establishing secure booting and long-term identity keys for SSL/TLS communication.

A MATTER OF TRUST

The TPM9670 is a member of a family of Trusted Platform Modules that are fully Trusted Computing Group (TCG) compliant with CC (EAL4+) and FIPS certification. Those modules are built around the Infineon OPTIGA TPM SLx 967y TPM2.0 chips that provide a secured key store and support a variety of encryption algorithms. The modules can thus establish robust protection for critical data, secure boots, and device identification, and entitle the Pi to become a root of trust.

The TPM9670 module comes with an IDC female socket for connecting it to the Raspberry Pi's expansion header and making use of its GPIOs and SPI terminals (see **Figure 1**). Application notes from Infineon state that the Pi should run a kernel version 4.14 or higher to support the operation of the TPM, so I did my investigation based on Linux 11 (Bullseye) with kernel 6.1.21-v8+. The SPI interface needs to be enabled in the Pi's configuration (`raspi-config`) for successful integration of the TPM module, resulting in the creation of two new devices identified as `/dev/tpm0` and `/dev/tpmrm0`. Fitting the TPM to the Raspberry Pi is shown in **Figure 2**.

Working with the TPM9670 can be achieved using either the Infineon ELTT2 or the TSS. Installing the toolbox requires building the code as instructed in GitHub [1], and here I put it simply in three shortcut steps (see `script1.sh`) *(Editor's Note: script1.sh and all files associated with this article are available for download at circuitcellar.com)*:

```
git clone https://github.com/
  Infineon/eltt2.git
cd eltt2
gcc -Wall -Wextra -std=c99 -g
  eltt2.c -o eltt2
```

TRUST SOFTWARE

The toolbox enables testing, diagnosing, and carrying out simple encryption operations using the command line. However, it requires disabling the software stack, such as the second option described in the following section, to acquire control of the TPM device. In **Table 1** I demonstrate some toolbox functions preceded by a stop command for the TPM stack service. The outcome of executing the toolbox commands is shown in **Figures 3-6** with superuser privileges instated, for example, `sudo`, due to hardware access restrictions for normal users.

Installing the software stack includes building six sub-components: the TPM Software Stack 2.0 (`tpm2-tss`) itself; the TPM2 Access Broker and Resource Manager (`tpm2-tabrmd`); TPM2 Tools (`tpm2-tools`); the TPM2 TSS Engine (`tpm2-tss-engine`); the Cryptsetup (`cryptsetup`); and the TPM2 PKCS#11 (`tpm2-pkcs11`).

The `tpm2-tss` consists of Application Programming Interfaces (APIs) to communicate with the TPM's different abstraction layers. The `tpm2-tabrmd` acts as a virtual memory manager responsible for swapping objects, sessions, and sequences in and out of the TPM limited memory as needed.

The `tpm2-tools` are command-line tools of the TSS that use the Feature API (FAPI), providing commands that start with "`tpm2_`" ; and Enhanced System API (ESAPI), providing commands that start with "`tss2_`", to communicate with the TPM or the Resource Manager. The `tpm2-tss-engine` implements a cryptographic engine for OpenSSL for the TPM2.0 using native TPM functionalities, and provides commands that start with "`openssl`".

The Cryptsetup adds TPM2.0 support for disk encryption and provides commands that start with "`cryptsetup`". The `tpm2-pkcs11` uses the TPM2.0 device as cryptographic token according to the Public-Key Cryptography Standard PKCS#11 and provides commands that start with "`pkcs11_tool`".

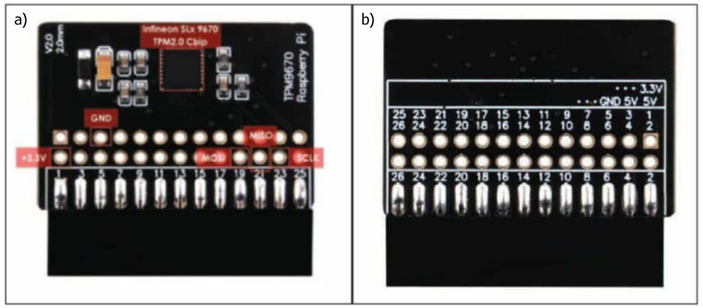


FIGURE 1
The TPM9670 module comes with an IDC female socket for connecting it to the Raspberry Pi's expansion header and making use of its GPIOs and SPI terminals.

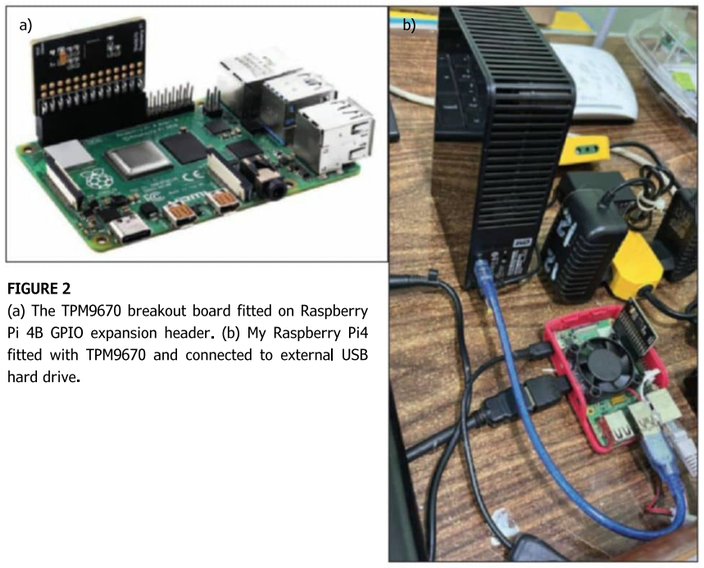


FIGURE 2
(a) The TPM9670 breakout board fitted on Raspberry Pi 4B GPIO expansion header. (b) My Raspberry Pi4 fitted with TPM9670 and connected to external USB hard drive.

Command Line	Effect
sudo systemctl stop tpm2-abrmd.service	Stop TPM2 stack service
sudo ./eltt2 -g	Show capability of fixed properties
sudo ./eltt2 -v	Show capability of variable properties
sudo ./eltt2 -t	Show timing details of some TPM activities
sudo ./eltt2 -a sha1 12345678	Hash a sequence of data bytes

TABLE 1
Some toolbox command lines demonstrate the toolbox usage.

```
pi@vpn-tpm:~/eltt2 $ sudo ./eltt2 -g

TPM capability information of fixed properties:
=================================================
TPM_PT_FAMILY_INDICATOR:        2.0
TPM_PT_LEVEL:                   0
TPM_PT_REVISION:                138
TPM_PT_DAY_OF_YEAR:             8
TPM_PT_YEAR:                    2018
TPM_PT_MANUFACTURER:            IFX
TPM_PT_VENDOR_STRING:           SLB9670
TPM_PT_VENDOR_TPM_TYPE:         0
TPM_PT_FIRMWARE_VERSION:        7.85.4555.0

TPM_PT_MEMORY:
=================================================
Shared RAM:                     0 CLEAR
Shared NV:                      1 SET
Object Copied To Ram:           1 SET

TPM_PT_PERMANENT:
=================================================
Owner Auth Set:                 0 CLEAR
Sendorsement Auth Set:          0 CLEAR
Lockout Auth Set:               0 CLEAR
Disable Clear:                  0 CLEAR
In Lockout:                     0 CLEAR
TPM Generated EPS:              0 CLEAR
```

FIGURE 3
The memory section determines if the TPM has access to different memory areas of the system and how they are managed.

```
pi@vpn-tpm:~/eltt2 $ sudo ./eltt2 -v

TPM capability information of variable properties:

TPM_PT_STARTUP_CLEAR:
=================================================
Ph Enable:                      1 SET
Sh Enable:                      1 SET
Eh Enable:                      1 SET
Orderly:                        1 SET
```

FIGURE 4
The TPM configurable properties that refer to the Platform Hierarchy, Storage Hierarchy, and Endorsement Hierarchy, respectively, for managing cryptographic keys and related operations. Orderly setting refers to commands' parameters that require protection in transit and storage.

```
pi@vpn-tpm:~/eltt2 $ sudo ./eltt2 -c

Clock info:
=================================================
Time since the last TPM_Init:
2460485 ms  =  0 y, 0 d, 0 h, 41 min, 0 s, 485 ms

Time during which the TPM has been powered:
31756961 ms  =  0 y, 0 d, 8 h, 49 min, 16 s, 961 ms

TPM Reset since the last TPM2_Clear:            8
Number of times that TPM2_Shutdown:             0
Safe:                                           1 = Yes
```

FIGURE 5
Clock information of TPM operation, including its initialization, uptime since recent power-on, and the count of shutdown attempts.

SIGN AND VERIFY DATA

The first step is to use the `tpm2-pkcs11` library as a module for the `pkcs11-tool`. The alias `pkcs11_tool` is generated for use in the signing process.

```
export mod_path=/home/pi/tssSetup/tpm2-
  pkcs11/src/.libs/libtpm2_pkcs11.so
alias pkcs11_tool='pkcs11-tool --module $mod_
  path'
```

Now, initialize a key store and create slot#1 in the TPM, and create a token for it with user pin `myfirstuserpin`.

```
pkcs11_tool --init-token --slot=1
  --label=myfirsttoken --so-pin=myfirstsopin
pkcs11_tool --init-pin --slot=1 --login
  --login-type="so" --so-pin=myfirstsopin
  --pin=myfirstuserpin
```

Then create a key pair labeled `mysignkey` for the signing process using the user pin.

```
pkcs11_tool --keypairgen --slot=1
  --label="mysignkey" --login
  --pin=myfirstuserpin --usage-sign \
--key-type=EC:prime256v1
```

Now you can sign `mydata.txt`, and generate the signature file `sign_data.sig` which is a hash digest stream of bytes encrypted with the private part of `mysignkey`.

```
pkcs11_tool --sign --slot=1
  --label="mysignkey" --pin=myfirstuserpin
  --mechanism=ECDSA-SHA1 \
--signature-format=sequence --input-
  file=mydata.txt > sign_data.sig
```

Furthermore, you need to extract the public part of `mysignkey`, which I called `pubkey.der`, out of the TPM and encode it into `pem` format and send it to the recipient of the data file.

```
pkcs11_tool --read-object --slot=1
  --label="mysignkey" --pin=myfirstuserpin
  --type pubkey > pubkey.der
openssl ec -inform DER -outform PEM -in
  pubkey.der -pubin > pubkey.pem
```

The signing process steps can be run as a shell script that I named `script2.sh`.

The recipient can verify the authenticity of the data file by applying the `openssl dgst` function which decrypts the signature file `sign_data.sig` using the public key `pubkey.pem` and compares the outcome with its generated digest of the data file. When both digests match, then the authenticity of the data file is verified.

```
openssl dgst -sha1 -verify pubkey.pem
  -signature sign_data.sig mydata.txt
```

Figure 7 and **Figure 8** illustrate the execution of the steps mentioned earlier on the command shell of the Raspberry Pi.

FILE ENCRYPTION AND DECRYPTION

Now you can attempt to encrypt a data file generated with the name [`mydata.txt`]. First, generate an RSA cypher key pair, named key.pair, and protect it with access passphrase `abc123` using openssl rsa.

```
echo "some secret text" > mydata.txt
tpm2tss-genkey -a rsa -s 2048 -p abc123 key.
  pair
openssl rsa -engine tpm2tss -inform engine
  -in key.pair -pubout -outform pem -out key.
  pub engine "tpm2tss" set
```

Then use `openssl pkeyutl` with `-encrypt` argument to encrypt the data file using the key.pair.

```
openssl pkeyutl -pubin -inkey key.pub -in
  mydata.txt -encrypt -out encrypted.data
hexdump -C encrypted.data
```

The file encryption steps can be run as a shell script such as the one I created as `script3.sh`

To decrypt the encrypted data file, use `openssl pkeyutl` again, but with the `-decrypt` argument, and respond to the passphrase challenge with `abc123` to get the original data file contents. **Figure 9** illustrates the command shell sequence of those steps on the Raspberry Pi. Note the difference in size of the original data file and its encrypted version: 16 to 256 bytes (see `script4.sh`).

```
pi@vpn-tpm:~/eltt2 $ sudo ./eltt2 -a sha1 12345678

TPM2_HashSequenceStart of '12345678' with SHA-1:
TPM Response:
80 01                           TPM TAG
00 00 00 0E                     RESPONSE SIZE
00 00 00 00                     RETURN CODE
 Command-specific response Data:
80 00 00 00

TPM2_SequenceUpdate:
TPM Response:
80 02                           TPM TAG
00 00 00 13                     RESPONSE SIZE
00 00 00 00                     RETURN CODE
 Command-specific response Data:
00 00 00 00 00 00 01 00 00

TPM2_SequenceComplete:
TPM Response:
80 02                           TPM TAG
00 00 00 31                     RESPONSE SIZE
00 00 00 00                     RETURN CODE
 Command-specific response Data:
00 00 00 1E 00 14 9B CE 73 D0
C8 B9 EC A4 F2 41 54 F3 BD 3B
8A A4 73 B1 C3 A9 80 24 40 00
00 07 00 00 00 00 01 00 00

Hash value extracted from TPM response:

0x00000000:   9B  CE  73  D0  C8  B9  EC  A4
0x00000008:   F2  41  54  F3  BD  3B  8A  A4
0x00000010:   73  B1  C3  A9
```

FIGURE 6
The hash value presented in 20 bytes for the 8 bytes of data preceded with TPM response for the hashing steps: start; sequence update; and sequence completion.

```
pi@vpn-tpm:~ $ pkcs11_tool --init-token --slot=1 --label=myfirsttoken \
--so-pin=myfirstsopin
Token successfully initialized
pi@vpn-tpm:~ $ pkcs11_tool --init-pin --slot=1 --login --login-type="so" \
--so-pin=myfirstsopin --pin=myfirstuserpin
User PIN successfully initialized
pi@vpn-tpm:~ $ pkcs11_tool --keypairgen --slot=1 --label="mysignkey" --login \
--pin=myfirstuserpin --usage-sign --key-type=EC:prime256v1
Key pair generated:
Private Key Object; EC
  label:      mysignkey
  Usage:      decrypt, sign
  Access:     sensitive, always sensitive, never extractable, local
  Allowed mechanisms: ECDSA,ECDSA-SHA1,ECDSA-SHA256,ECDSA-SHA384,ECDSA-SHA512
Public Key Object; EC  EC_POINT 256 bits
  EC_POINT:   044104a5ff1d642416afc1601f1a3d13f495e9744081846cff3ad0f76fa6ad85f972af98816a6243b17ba023056d
6dcb23943926a13383e66a5d1f9148d332bd35a80d
  EC_PARAMS:  06082a8648ce3d030107
  label:      mysignkey
  Usage:      encrypt, verify
  Access:     local
pi@vpn-tpm:~ $ pkcs11_tool --list-objects --slot=1
Public Key Object; EC  EC_POINT 256 bits
  EC_POINT:   044104a5ff1d642416afc1601f1a3d13f495e9744081846cff3ad0f76fa6ad85f972af98816a6243b17ba023056d
6dcb23943926a13383e66a5d1f9148d332bd35a80d
  EC_PARAMS:  06082a8648ce3d030107
  label:      mysignkey
  Usage:      encrypt, verify
  Access:     local
pi@vpn-tpm:~ $ pkcs11_tool --sign --slot=1 --label="mysignkey" --pin=myfirstuserpin --mechanism=ECDSA-SHA1
 --signature-format=sequence --input-file=mydata.txt > sign_data.sig
Using signature algorithm ECDSA-SHA1
```

FIGURE 7
Initializing a key store in the TPM module, generating signing key pair within the TPM, and signing my data file.

```
sudo openssl pkeyutl -engine tpm2tss
   -keyform engine -inkey key.pair -decrypt
   -in encrypted.data \
   -out decrypted.data engine “tpm2tss” set
hexdump -C decrypted.data
```

DISK ENCRYPTION AND DECRYPTION

First, generate an encryption key for the LUKS volume and name it `luks.key`.

```
dd if=/dev/urandom of=luks.key bs=1 count=64
```

Seal that key into the TPM, something done in several steps: create a seal file called `primary.ctx`, which is used as a parent for internally created TPM public and private keys, for example, `key obj.pub` and `obj.priv`; load the key hierarchy into the TPM, a process which produces a load file, for example, `sealed.ctx`, that is needed later on for unsealing the volume encryption key (see **Figure 10**).

```
tpm2_createprimary -C o -c primary.ctx
tpm2_create -C primary.ctx -i luks.key -u
   obj.pub -r obj.priv
tpm2_load -C primary.ctx -u obj.pub -r obj.
   priv -c sealed.ctx
```

Now commence encrypting the LUKS volume using the `luks.key` prior to deleting it from the Pi file system (see **Figure 11**).

```
sudo cryptsetup luksFormat /dev/sda3 --key-
   file luks.key
```

Upon successfully completing the sealing process and volume encryption, securely delete the volume encryption key file (see `script5.sh`).

```
shred -zu luks.key
```

The disk volume is now configured as a LUKS volume, and has no file system yet to make it accessible to the Pi OS. To create the file system, you need to unseal the encryption key using the seal load file and produce the volume encryption key again, which I called `luks.unsealed`. Now, decrypt the LUKS volume with the key, producing a new device called `/dev/mapper/my_volume`. Optionally, you could use a custom script to unseal the key from TPM and unlock the LUKS volume at boot time (see `script6.sh`).

```
tpm2_unseal -c sealed.ctx -o luks.unsealed
```

```
sudo cryptsetup luksOpen /dev/sda3 my_volume
   --key-file= luks.unsealed
```

Now, create an EXT4 file system on that device (see **Figure 12**) and mount it on a directory of your choice, such as. `/media/WD3`. Here I accessed the EXT4-mounted volume as a regular partition on the hard drive (see `script7.sh`).

```
sudo mkfs.ext4 /dev/mapper/my_volume
sudo mount /dev/mapper/my_volume /media/WD3
sudo touch /media/WD3/mydata.txt
sudo nano /media/WD3/mydata.txt
```

```
pi@vpn-tpm:~ $ pkcs11_tool --read-object --slot=1 --label="mysignkey" --pin=myfirstuserpin --type pubkey >
 pubkey.der
pi@vpn-tpm:~ $ openssl ec -inform DER -outform PEM -in pubkey.der -pubin > pubkey.pem
read EC key
writing EC key
pi@vpn-tpm:~ $ openssl dgst -sha1 -verify pubkey.pem -signature sign_data.sig mydata.txt
Verified OK
```

FIGURE 8
Extracting the signing public key out of the TPM module, encoding it into pem format, and verifying my data file authenticity.

```
pi@vpn-tpm:~ $ echo "some secret text" > mydata.txt
pi@vpn-tpm:~ $ tpm2tss-genkey -a rsa -s 2048 -p abc123 key.pair
pi@vpn-tpm:~ $ openssl rsa -engine tpm2tss -inform engine -in key.pair -pubout -outform pem -out key.pub
engine "tpm2tss" set.
Enter password for user key:
writing RSA key
pi@vpn-tpm:~ $ openssl pkeyutl -pubin -inkey key.pub -in mydata.txt -encrypt -out encrypted.data
pi@vpn-tpm:~ $ hexdump -C encrypted.data
00000000  57 d6 2d fd bf 8f b9 0f  6d 95 a4 b2 3d 7e 94 cb  |W.-.....m...=~..|
00000010  8b 80 62 52 bc 7a 77 5b  05 9a aa 9b 88 17 ca e7  |..bR.zw[........|
00000020  92 1c f9 58 50 79 7e bc  3d a5 be bf 40 fe b6 99  |...XPy~.=...@...|
00000030  85 d0 b9 b2 c3 ae d8 da  43 e6 c6 e1 22 75 87 ca  |........C..."u..|
00000040  e0 14 f4 e0 30 33 8e ac  96 ae f3 f5 f4 2a 06 61  |....03.......*.a|
00000050  b3 03 34 f7 7b d3 6c b8  61 62 1c 92 d0 2d 6c 9e  |..4.{.l.ab...-l.|
00000060  d4 b8 9c 71 d0 d8 c7 e3  94 e4 4a 63 d6 11 56 28  |...q......Jc..V(|
00000070  c0 d6 dc 1d c3 ca b0 40  1e f1 0e 36 db c0 04 f9  |.......@...6....|
00000080  50 c5 86 15 59 bf 5e e4  93 ee 8d 0f 7d ee e8 a9  |P...Y.^.....}...|
00000090  6e f7 ab c7 8e 10 08 8a  2d 1a b6 5d 82 bf 7b 9f  |n.......-..]..{.|
000000a0  22 7d 41 8f 4b a0 b0 27  7f 76 80 bc 94 34 a5 8b  |"}A.K..'.v...4..|
000000b0  e7 1e 9d 4e 22 ed 9a 55  42 6c 80 58 e6 30 32 17  |...N"..UBl.X.02.|
000000c0  c0 a6 d1 fd ca cf eb 9d  49 00 36 e7 09 18 66 66  |........I.6...ff|
000000d0  52 45 dd 87 1d 36 df 79  82 e9 f6 14 b6 9b 7a 24  |RE...6.y......z$|
000000e0  60 05 bb f1 12 de 92 61  26 b4 62 c6 24 31 d3 48  |`.......a&.b.$1.H|
000000f0  b9 3d 5b 7d 33 33 72 5d  d8 ff 5a ad 77 e7 2a 1b  |.=[}33r]..Z.w.*.|
00000100
pi@vpn-tpm:~ $ sudo openssl pkeyutl -engine tpm2tss -keyform engine -inkey key.pair -decrypt -in encrypte
d.data -out decrypted.data
engine "tpm2tss" set.
Enter password for user key:
pi@vpn-tpm:~ $ hexdump -C decrypted.data
00000000  73 6f 6d 65 20 73 65 63  72 65 74 20 74 65 78 74  |some secret text|
00000010  0a                                                |.|
00000011
pi@vpn-tpm:~ $
```

FIGURE 9
Data file encryption and decryption procedure using TPM2-TSS and openssl.

```
pi@vpn-tpm:~ $ tpm2_createprimary -C o -c primary.ctx
name-alg:
  value: sha256
  raw: 0xb
attributes:
  value: fixedtpm|fixedparent|sensitivedataorigin|userwithauth|restricted|decryp
t
  raw: 0x30072
type:
  value: rsa
  raw: 0x1
exponent: 0x0
bits: 2048
scheme:
  value: null
  raw: 0x10
scheme-halg:
  value: (null)
  raw: 0x0
sym-alg:
  value: aes
  raw: 0x6
sym-mode:
  value: cfb
  raw: 0x43
sym-keybits: 128
rsa: 9661d8795fbfbffbab06a6b2ce1ea1c8e504e01e443dd44367b99d02a9f4430ca857de0a4ae
a9f5cc24513e459f78e240f8060d1ad61f2e973ced696e8a381e4adf4b2c650f3d6fbce67297fb02
0cbd16c9cf14150cf35318c45fc77f64f90d7b41735640d699554e6fdd86dbfffeb0b975029139f6
ad49815b98650e9d89050633b9a70d0e3dd8d60d9c73540e88ae0ebde3f57cf08dbf1e9f2f5390fe
0d1dbf4993c87b28163edc25fde113df8ad092f6e059166fef4d0ad4633b8e36a17871bc79960452
f8e5f9e3878adb2a06402451472b7bef947ee2bb0715e2c596f986cd0112858eb57f3330638b2c9a
f5a2386addff1acc8795f351d7e2e8766e0a7
pi@vpn-tpm:~ $ tpm2_create -C primary.ctx -i luks.key -u obj.pub -r obj.priv
name-alg:
  value: sha256
  raw: 0xb
attributes:
  value: fixedtpm|fixedparent|userwithauth
  raw: 0x52
type:
  value: keyedhash
  raw: 0x8
algorithm:
  value: null
  raw: 0x10
keyedhash: 98a1413c85663c105eddba839d2504eff4c7ec94f69ce5e1417ced486e405d32
pi@vpn-tpm:~ $ tpm2_load -C primary.ctx -u obj.pub -r obj.priv -c sealed.ctx
name: 000b74d1ddc74f5bba6beadf8a69fad0159f90b77b88a0ea301d17188cfec6ff81a0
```

FIGURE 10
Formatting encrypted volume with LUKS format.

FEATURES

FIGURE 11
Creating EXT4 file system on LUKS volume.

```
pi@vpn-tpm:~ $ sudo cryptsetup luksFormat /dev/sda3 --key-file luks.key
WARNING: Device /dev/sda3 already contains a 'ext4' superblock signature.

WARNING!
========
This will overwrite data on /dev/sda3 irrevocably.

Are you sure? (Type 'yes' in capital letters): YES
```

FIGURE 12
The encrypted file system is created and mounted, where it can be used like any other file system.

```
pi@vpn-tpm:~ $ sudo mkfs.ext4 /dev/mapper/my_volume
mke2fs 1.46.2 (28-Feb-2021)
Creating filesystem with 12795904 4k blocks and 3203072 inodes
Filesystem UUID: 8f2c0fb2-2754-4988-a51a-164994146c91
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,
        4096000, 7962624, 11239424

Allocating group tables: done
Writing inode tables: done
Creating journal (65536 blocks):
done
Writing superblocks and filesystem accounting information: done
```

After the completion of usage, the LUKS volume needs to be encrypted again, and this is done in two steps. First, unmount the mapper device, for example., `/dev/mapper/my_volume`. Second, do a `luksClose` to re-encrypt it (see `script8.sh`).

```
sudo umount /dev/mapper/my_volume
sudo cryptsetup luksClose my_volume
```

TPM SECURITY VULNERABILITY

BitLocker is Microsoft's disk encryption feature designed to protect data at rest on disk volumes. It has been an integral feature of the Windows operating system for decades. BitLocker can utilize TPM modules to strengthen its security. At startup time, BitLocker looks for the cryptographic keys in the TPM and uses them to decrypt the boot volume, thus enabling Windows to load and function.

Similar to the BIOS ROM, the TPM is connected to the Low Pin Count (LPC) bus of IBM-compatible computers, which enables low-bandwidth devices to communicate with the CPU. Such connectivity introduced vulnerability to BitLocker, since the exchange of encryption keys with the TPM was susceptible to eavesdropping using external tools wired to the LPC.

Some of the reported tools are the iCEstick FPGA Tool, and Raspberry Pi Pico. Having said that, we can see how easy it is to apply the previous techniques to the SPI interface of Raspberry Pi through its exposed expansion header. To overcome such vulnerability, the Pi's SD card and the expansion header should be protected against unauthorized physical access and tampering. This can be achieved by using tamper-proof cases and tamper-detection mechanisms that alert the owner if the case is opened.

SECURITY BUILT IN

Trusted Platform Modules add another layer of hardware security to single-board computers such as Raspberry Pi. You've now seen the TPM9670 application for basic encryption operations, digital signing and verifying data, and file and disk encryption and decryption. TPM9670 supports popular cryptographic algorithms, provides secure storage of cryptographic keys, and performs encryption and decryption operations internally, and thus enforces data confidentiality and authenticity.

In Part 2 of this article series, I will present the TPM application for secure boot and identity to establish a root of trust. Together, the two TPM applications can make single-board computers and the applications they power much more secure.

ABOUT THE AUTHOR

Gamal Labib has been an enthusiast of embedded systems for two decades and currently a mentor at codementor.io. He holds a MEng and a PhD in IT. Besides writing for technical magazines, he is a visiting associate professor at Egyptian universities, and a certified IT consultant.

Email: drgamallabib@yahoo.co.uk

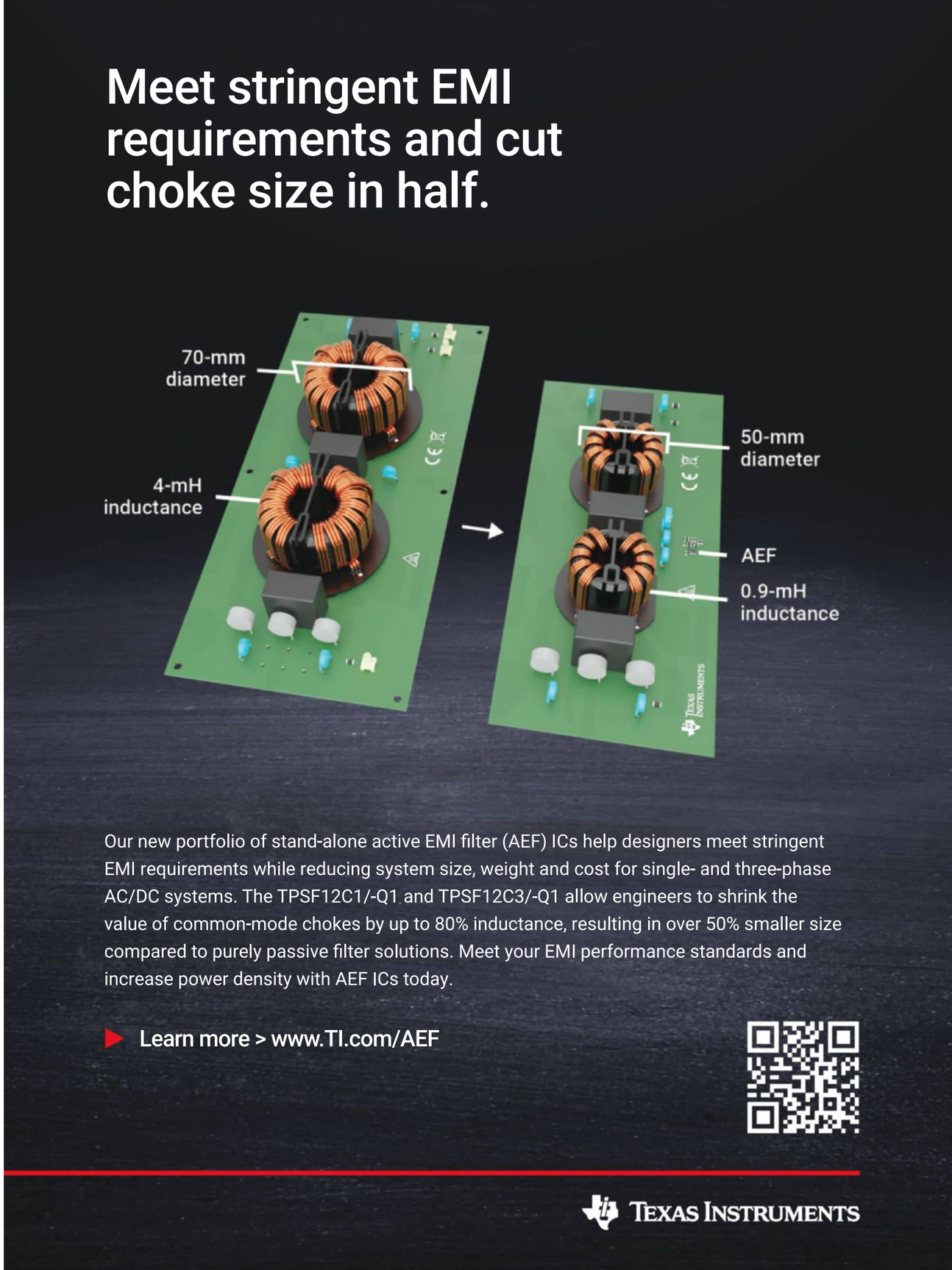
Additional materials from the author are available at: www.circuitcellar.com/article-materials

References [1] as marked in the article can be found there.

RESOURCES

ENTRUST | www.entrust.com

Infineon | www.infineon.com


Meet stringent EMI requirements and cut choke size in half.
70-mm diameter
4-mH inductance
50-mm diameter
AEF
0.9-mH inductance
Our new portfolio of stand-alone active EMI filter (AEF) ICs help designers meet stringent EMI requirements while reducing system size, weight and cost for single- and three-phase AC/DC systems. The TPSF12C1/-Q1 and TPSF12C3/-Q1 allow engineers to shrink the value of common-mode chokes by up to 80% inductance, resulting in over 50% smaller size compared to purely passive filter solutions. Meet your EMI performance standards and increase power density with AEF ICs today.
Learn more > www.TI.com/AEF
Texas Instruments

Synth Hero: Where Synth Legends Are Made

Creating a Retro Game Using a Raspberry Pi RP2040 MCU

By
Chukwuemeka Adebi and Sergio Diaz

A novel rhythm game—a cross between Piano Tiles and Guitar Hero—created by two Cornell University undergraduates, combines interests in gamified music and electronics prototyping with a project that has lessons for many makers.

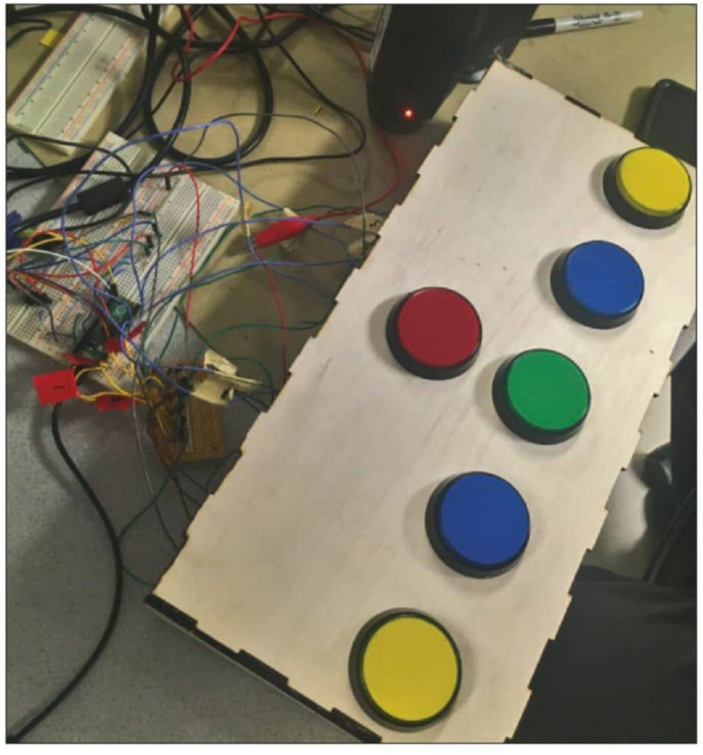


FIGURE 1
The physical apparatus of the game. This includes the laser-cut frame, solder board, and breadboard with wire connections. See also Figure 5.

We started this project with the intention of making a game that could teach people to play an instrument using songs and a button system. The project was inspired by rhythm games such as Piano Tiles and Guitar Hero. Our interface looks like a simplified version of these games, with tiles (called "NoteBars") that progress down the screen. When you press one of the buttons, it plays a corresponding note. The goal is to press the notes of each song correctly at the right time, until the song ends. Correctly pressing a note increases your score, and the goal is to get the highest score possible. A player knows when the button needs to be pressed because the tile overlaps with the score line on the Video Graphic Array (VGA) screen, and the button illuminates.

USER EXPERIENCE

The user plays the game with the physical apparatus shown in **Figure 1**, which is composed of five LED pushbuttons. When the game begins, a menu screen appears. The user navigates through this menu with the middle green and red buttons on a physical game board (Figure 1). The green button cycles through each option, and the red button selects the option highlighted.

From the main menu, the user can access the difficulty, song selected, or leave the game. When the user eventually selects "Start Game," the song starts and NoteBars begin to flow down the screen, until either the user presses the red button—which pauses the game—or the song ends.

During the song, the user must press the button corresponding to each NoteBar as it traverses a black bar at the bottom of the screen. If the user times a press correctly, the NoteBar turns cyan, and the score gets incremented. Otherwise, the NoteBar turns red. Points are added to the score when it is correctly pressed. When the song is over, the device displays the player's score and provides an option to return to the main menu.

HIGH LEVEL DESIGN

A block diagram of the game's hardware is shown in **Figure 1**. The game was built using a Raspberry Pi RP2040 microcontroller. The RP2040 has two cores, designated Core 0 and Core 1. We utilized both cores to control the gameplay: Core 0 is responsible for receiving signals from the buttons, controlling the game logic, and displaying the graphics on the VGA screen as indicated in **Figure 2**; while Core 1 is responsible for synthesizing all audio using Direct Digital Synthesis (DDS), and outputting it to the digital-to-analog converter (DAC). We did not face any trade-offs on the software aspect, but we did experience some issues, which are explained in more depth in the software section of this article.

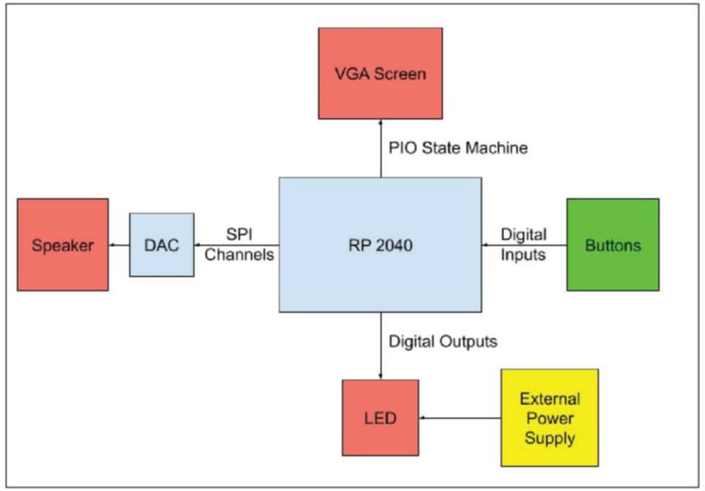


FIGURE 2
Block diagram of the hardware connections of the physical apparatus. This image also includes the types of connections.

EQUATIONS USED FOR NOTE SOUNDS

Pure sine waves are easy to generate using DDS, but they are a bit boring to listen to. We wanted to generate a synthetic sound that was more than a simple sine wave, allowed for clearly distinguishable tones, and was smooth enough to almost sound like a synth bell. We accomplished this through additive synthesis of four sine waves, with experimentally determined frequencies and amplitudes. Equation 1 was used for our waveform, and **Figure 3** shows each of the constituent waveforms (in green, blue, and purple) and the additively synthesized waveform in red. The red line in the graph signifies the full equation, while the others are the equation decomposed into the sinusoidal additions of the equation.

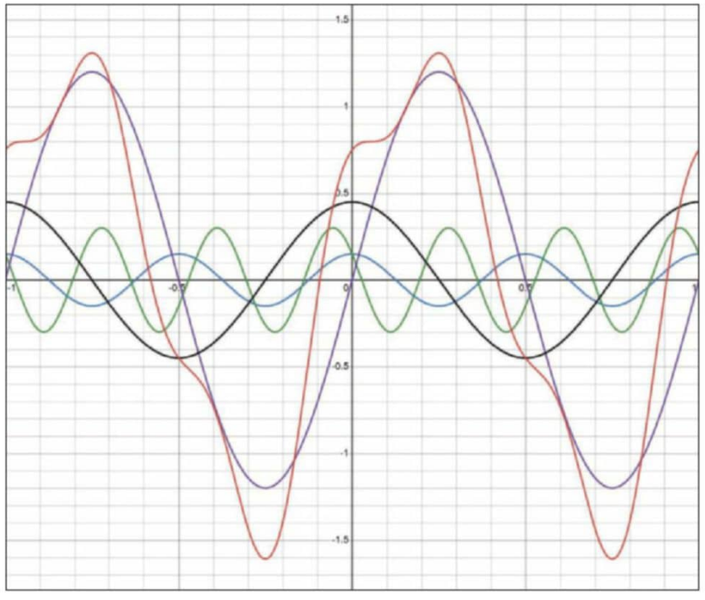


FIGURE 3
Graph of additive synthesis of four sine waves. Red is the color of all the synthesized additive sine waves and the other three are the decomposed sine waves.

$$f(x) = 0.15\sin\left(4\pi x + \frac{\pi}{2}\right) - 0.3\sin\left(6\pi x - \frac{\pi}{6}\right) + 1.2\sin(2\pi x) + .45\sin\left(2\pi x + \frac{\pi}{4}\right)$$

SOFTWARE DESIGN

Main Threads/Interrupts: The program has three main operations: the main game loop (Core 0), the button interrupts (Core 0), and the audio synthesis interrupts (Core 1). The game loop is a thread. Threads are used to run concurrent programs in our build. Each of the main operations is run on a different thread.

Button Interrupts: We introduced the button interrupts, because we experienced a particularly unique problem when working with our arcade-style buttons. The problem was the debouncing of buttons. These buttons bounced for a considerable length of time each time the user pressed or released them. This meant we could not just check the General Purpose Input/Output (GPIO) pin for each button, because its GPIO output wasn't necessarily indicative of its quiescent state while bouncing.

To resolve this issue, we created a timer interrupt that runs once every 900µs and enforces a maximum frequency with which the button can change state. It does so by maintaining and updating two state variables associated with each button.

Each button has a "repeats" counter and a "signals" signal. The repeats counter calculates for how many interrupts the button is detected to be in the OFF state. If the repeat counter is less than 10, then it isn't ready to be clicked again. However, if it reaches 10, it can be clicked.

When a button is clicked, its repeat counter is set to 0 and its signal is set to 1. This signal variable is essential, since button clicks are being recorded asynchronously, and the rest of the program needs to know when a button has recently been clicked. So, when the signal for a button is set to 1, this means that a button was recently clicked. Once the main thread reads this, it resets the button's signal to 0. An array of values is used to track button presses and releases, so when the button is released, it will automatically go back to the state of not being pressed.

Audio Synthesis Interrupts: To guarantee timing, we performed audio synthesis in a timer interrupt. We placed this interrupt on Core 1 to prevent it from interfering with game logic running on Core 0. Also, we wanted the frequency to sound smooth and have no interruption. To handle these issues, we utilized the FIFO that connects Core 0 to Core 1 in the RP2040.

We had used DDS in previous projects to generate pure sine waves continuously. However, we wanted to trigger these sounds to play when a specific action occurred (when a button was pressed). We did not want the sound to play continuously on a loop. Because the audio synthesis was on Core 1, we needed communication between Core 1 and Core 0.

To communicate between cores, we used FIFO READ and WRITE operations. Core 0's main thread detects a button signal, then it writes the frequency corresponding to this button into the FIFO. Core 1's main thread constantly reads from the FIFO, so it is able to detect that Core 0 wants it to play a specific frequency.

At this point, the thread adds data to a global array that instructs the audio synthesis interrupt to begin playing a synth sound at the specified frequency. The interrupt loops over this array to check what frequencies it needs to play. If it has any number of frequencies, it will play them all until they have been held for their total duration. If it doesn't have any frequencies, it will simply remain silent and begin to play when Core 0 provides it with some. This is how we were able to synthesize audio (even multiple notes at once) without affecting our performance.

The game loop thread was the most complex part of the project, since it handled a lot! The body of it manages the screen state—essentially, what screen is currently displayed. It switches among six main screens:

- Home/Menu
- Gameplay
- Paused
- Game-Over
- Song Selection
- Difficult Selection

Finite Game Screen States: We implemented a Finite State Machine (FSM) to handle all the screen states. A FSM of the screen state is shown in **Figure 4**. Screens such as Song Selection and Difficulty Selection weren't too difficult to implement, since they mainly just changed certain global variables (such as the data that controlled the song and difficulty for the next game). The most difficult screen to implement was Gameplay, shown in **Figure 5**.

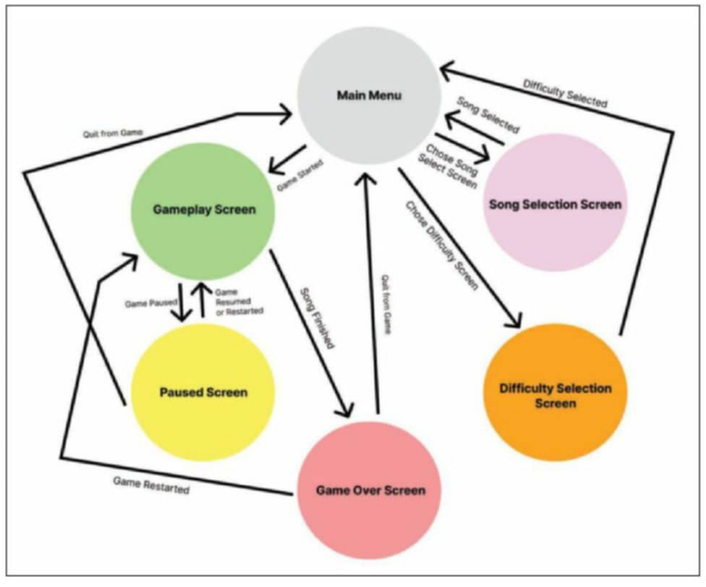


FIGURE 4
Diagram of our Finite State Machine (FSM), showing how our menu screens work.

```
// Struct which defines a NoteBar -- each bar that
represents a Note in game
        typedef struct NoteBar {
            int yPos;
            int numCol;
            struct NoteBar* prev;
            struct NoteBar* next;
            int clickedState;
            int passed;
            int flashed;
        } NoteBar;
```

LISTING 1
The C implementation of the NoteBar struct used for our game.

For connecting our apparatus to a monitor to show the game, we used a VGA screen (Figure 5) and a VGA library, which used Direct Memory Access (DMA) to send data to the VGA monitor without using up CPU cycles. As for the hardware aspect, we wired HSync and VSync directly from RP2040 ports, as well as the R, G, and B wires, which were connected to the VGA cable using 330Ω resistors.

NoteBars: The main element of the gameplay is the NoteBar struct. A NoteBar is the "object" that indicates when a player has to press a button to score points and play the correct note at the right time.

The code is shown in **Listing 1** The variables in the NoteBar, starting with `yPos`, is an integer value of the y position of a NoteBar on the screen. The variable `numCol` is the number of the column in which the NoteBar is located. The next two parameters, `prev` and `next` are pointers to the previous and next Notebars. They act as linked lists. The parameter `clickedState` determines whether the

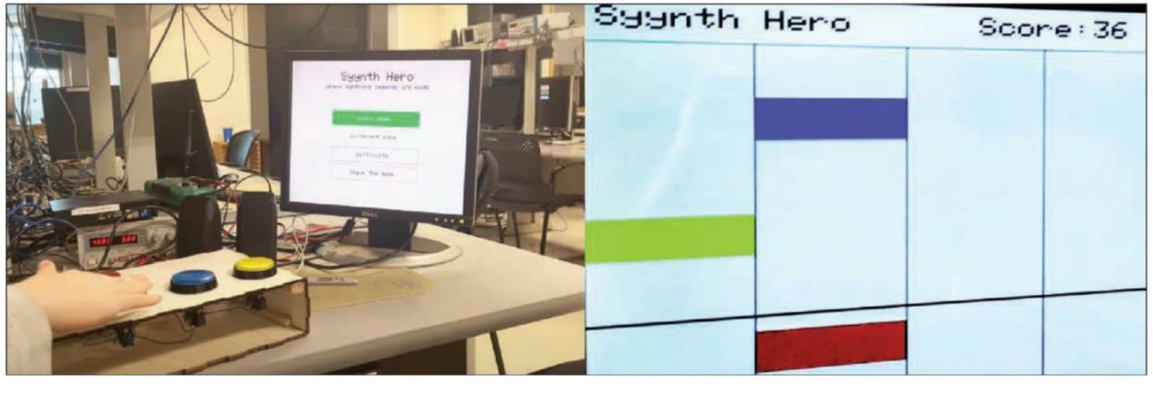


FIGURE 5
The opening menu screen with the physical apparatus, while a button is being pressed. The image next to it is the screen while in-game.

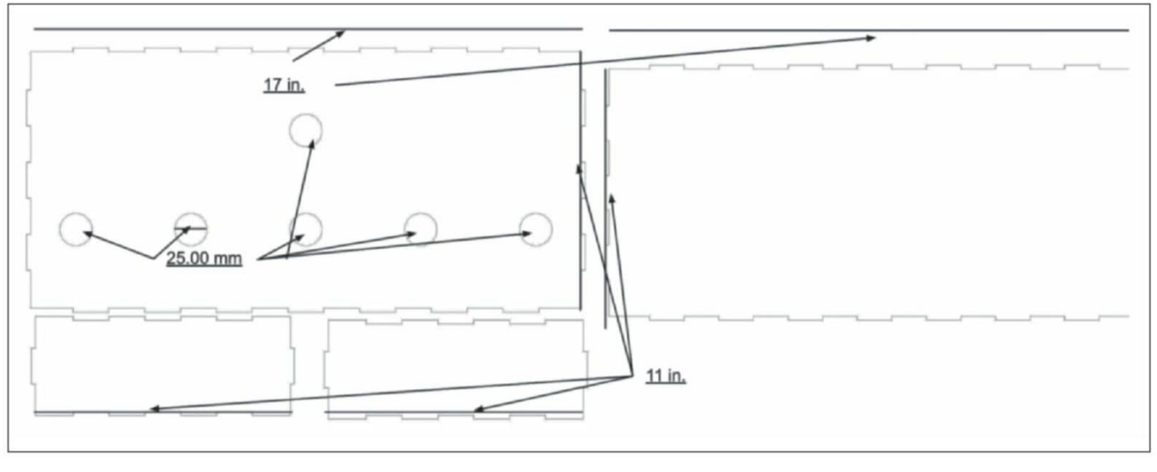


FIGURE 6
CAD file of the wooden frame created to hold the buttons and circuitry. The measurements are in inches and millimeters.

NoteBar has been clicked or not. The parameter `passed` determines whether the NoteBar has passed the score line or not. The final parameter of the NoteBar, `struct`, is the integer variable flashed. It determines whether the physical LED inside the button is on or not.

While many of the variables within the NoteBar are well-explained by the comments, the `next` and `prev` pointers may be less clear. We added these to allow NoteBars to act like a linked list, so that in each iteration of the game loop, the game loop can loop through all of the NoteBars on the screen, update their positions, and redraw them. This also allows the game to remove NoteBars with ease. The reason a linked list works with our implementation is that each NoteBar moves down the screen at a constant speed predetermined by the game's difficulty. Therefore, a NoteBar that is further up the screen should never disappear before a NoteBar below it does.

As mentioned, the implementation of the NoteBar is essential, because it allows us to easily loop through all the NoteBars and redraw them to produce a functional game. Aside from this, our gameplay contains functions to check if any buttons have been clicked and to send a message to Core 1 to play the appropriate sound. It also contains a yield to keep the frame rate constant. If the frame rate weren't constant, then rendering fewer NoteBars would result in much faster game play, and more NoteBars would result in much slower gameplay—which would make the song sound extremely inconsistent.

Encoding the Songs: The last section of our software was encoding the songs themselves. We needed four main pieces to code the songs:

- The column in which a NoteBar would appear
- The time a NoteBar would appear in such column
- The frequency that would be played when a specific button was hit
- The appropriate time to play such frequency

We needed three and four because we were able to put only five columns in the game, but most songs include more than five notes. So, if we wanted to play more complex songs, we'd need to include more frequencies. The code for encoding the songs, along with all the code for this project, can be found in the online listing at *CircuitCellar.com* that accompanies this article.

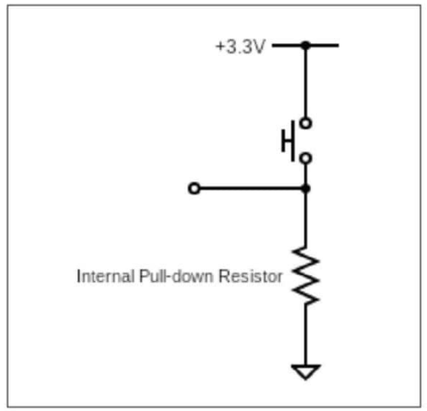


FIGURE 7
Schematic for each of the pushbuttons and how they are connected to a GPIO pin.

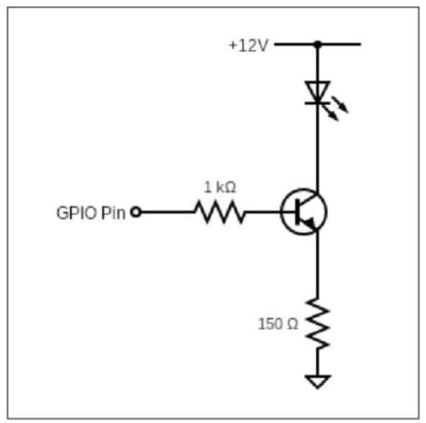


FIGURE 8
Schematic for each of the LEDs within the push buttons and their connections to GPIO pins.

HARDWARE DESIGN

Mechanical Structure: The mechanical structure of the game is a wooden box with an open front and open back, for easy access to parts for assembling and wiring. The block diagram of the hardware connections in the device is shown in Figure 2. The only things visible to the player are the buttons on the top of the board.

Frame CAD Schematic: The CAD model of the frame is shown in **Figure 6**. After the frame was cut, we installed the game's six, 60mm buttons. Each of the buttons connected to this wooden box, has four ports. Two of these ports are used for the button's output signal that inputs into the RP2040. One wire is connected to a GPIO pin, and the other wire is connected to ground. The GPIO pins have a built-in resistor and have a voltage of 3.3V. In the software, we pull up these GPIO pins so that a LOW signal indicates that the button has been pressed. **Figure 7** shows the button circuit.

The other two ports on the button control its LED, which is connected to a solderboard connected to a 12V power supply. Each LED is individually connected to a 2N3904 transistor. The bases of these transistors are then connected to GPIO pins through a 1kΩ resistor. The voltage supply is connected through the LED to the collector of the transistor. The emitter is connected through a 150Ω resistor to ground. **Figure 8** shows the schematic for the LEDs.

We also utilized the Serial Peripheral Interface (SPI) features to output audio that was generated via DDS. A MCP48X2 DAC was used to take in this SPI signal from the RP2040 and convert it from digital to analog. Both of the DAC's input channels were used, since we did generate some audio concurrently whenever a user pressed more than one button at a time. The output of the DAC was then connected to a 3.5mm audio socket (aux), which allowed a speaker to be plugged in to synthesize the audio.

A PLAYABLE GAME

We were pleased with the extent to which our final product, shown in Figure 5, matched our original vision. We never faced any issues after this extensive testing whenever we or others played the game. When other students played our game, we had to explain how the red and green center buttons allowed them to control the menu options. However, once they got the hang of it, they found the interface to be very intuitive.

We believe we were able to achieve the intended behaviors almost immediately, due to the safety mechanism we implemented. It required all major global variables that indicate screen and selection state to be controlled from within the main Core 0 thread. Doing this ensured that, whenever a user changed the screen's state, it would do so in the main thread. Also, the use of simple arcade buttons also provided some base level of safety, because it was impossible for a user to input NULL or other undesirable types into our program.

We believe we created an appealing project because our implementation was on par with our vision. But if we had to pick something to improve for the next iteration of the game, we would spend more time trying to synthesize different types of instruments other than a simple synth. We think it would be cool to allow the user to choose from a piano, guitar, flute, or other musical instrument. However, based on previous attempts, we understand that synthesizing these instruments might be a significant challenge, and is really feasible only with a longer-term project. Overall, we are very satisfied with the results of our game and think it is fun to play and share with our friends!

ABOUT THE AUTHORS

Chukwuemeka Adebi (cea57@cornell.edu) is currently a graduate student-athlete at Vanderbilt University, formerly a student-athlete at Cornell University. He is majoring in engineering management and on the varsity football team. His research interests are largely within computer architecture, embedded systems, operating systems, hardware acceleration, and security. Adebi is always open to new experiences and eager to learn.

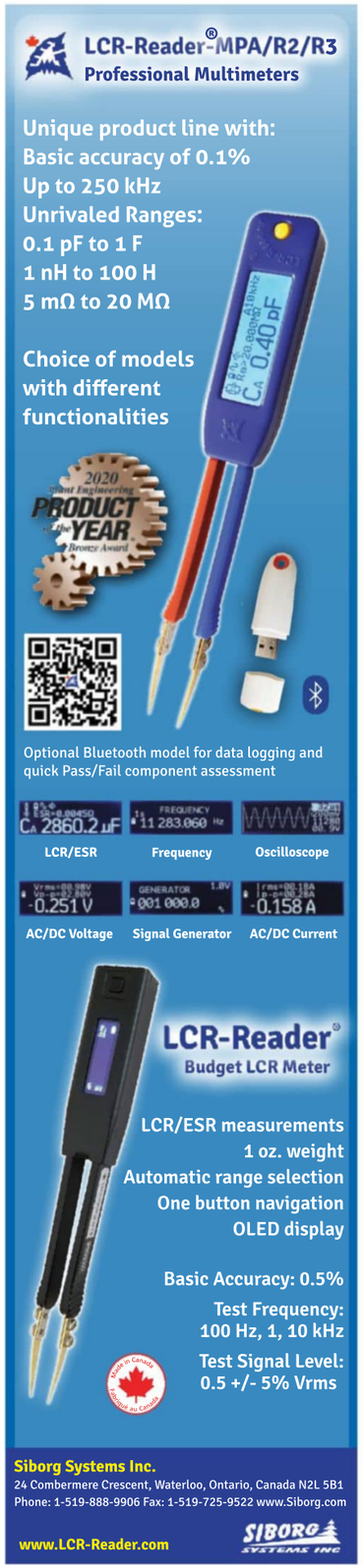
Sergio Diaz is an aeronautical engineer and airport construction manager in Barcelona, Spain. He has a passion for his profession, excellent organizational skills, and a flair for teamwork. Diaz is highly adaptable to all kinds of environments and goal oriented. He has great ambition to acquire new knowledge, to be able to put into practice and develop the knowledge already obtained.

Additional materials from the author are available at: www.circuitcellar.com/article-materials

References [1] to [3] as marked in the article can be found there.

RESOURCES

Raspberry Pi | www.raspberrypi.com



Mesh Networks

Reliable, Decentralized Communication Where Single Routers Fail

By
Brian Chee

Unlike conventional networks with a central router, mesh networks distribute data across multiple "nodes" that communicate directly with one another. In this article, Brian shares his experiences with these self-healing networks, in projects that ranged from getting battlefield information to commanders and determining why a Hawaiian plant was becoming extinct, to creating a home sensor network. He discusses in detail the features of Zigbee and LoRaWAN mesh networks.

The Defense Advanced Research Projects Agency (DARPA) is a research and development agency of the United States Department of Defense. Its mission is to prevent technological surprises by creating breakthroughs in technology for national security. It has been funding moon shots since the Soviet Union launched Sputnik into orbit in 1957 as the first man-made satellite to orbit the earth. Then-President Eisenhower created the Advanced Research Projects Agency (ARPA to bridge the perceived technological gap during the Cold War.

FIGURE 1
The Hawaii catchfly, Silene hawaiiensis, a rare and threatened species found only on the island of Hawaii.

THE SENSOR INFORMATION TECHNOLOGY (SENSIT) PROJECT

DARPA has fanned the innovation spark of researchers all over America to answer a simple question: "Can it be done?" One such question was whether a self-organizing, self-healing network could be created, so that war fighters could simply drop devices as they traveled to relay more timely and accurate data from the battlefield to area commanders. The project was called Sensor Information Technology (SensIT), and it funded research into this question at numerous research organizations, including the University of Hawaii's Information and Computer Sciences Department. The principal investigators were Edoardo Biagioni (Information and Computer Science), Kent "Kim" Bridges (Botany), and me (Advanced Network Computing Laboratory).

The base concept of mesh networking is the ability of each device or "node" in the network to relay network connections—like a "bucket brigade." Each node needs to keep track of all the upstream and downstream nodes, so that if a link should fail, the network could self heal by choosing an alternate path, and send updates to the other nodes in the mesh to "heal" the path to the primary node.

FIGURE 2
Andy Wheeler, one of the inventors of ZigBee who developed the system.

The final product of SenseIT was a ruggedized sensor package that would relay acoustic and seismic data over this mesh network that war fighters could simply drop along the way, and that would bring more accurate battlefield information to commanders.

THE VOLCANO NATIONAL PARKS (PODS) PROJECT

The PODs project at the University of Hawaii was carried out by the "tree hugger" contingent of the SensIT project and proposed a study area at Hawaii Volcano National Park. The idea was to create a system that would relay data out of deep forests, caves, or crevasses at the volcano area. In our case we wanted to collect data to find out why the Hawaiian catchfly, *Silene hawaiiensis,* (**Figure 1**) was going extinct. Our cooperative project with the MIT Media Lab had us testing a system [1] designed by Andy Wheeler of MIT (**Figure 2)**, combined with sensors and housings created by the University of Hawaii team. The rule was, "If you can't see it, it can't be there,"—that is, to disguise the nodes, since millions of visitors didn't come to the park to see bright orange instrument cases.

The goal was to collect high-resolution digital images of our plants throughout the day, combined with environmental data, including temperature, humidity, soil moisture, basic wind speed (can a seed be blown by the wind), and light levels. The sensors were hidden in artificial tree branches hidden in the Ohia trees (**Figure 3** and **Figure 4**), and fake rocks scattered along the Chain of Craters Road from the Jager Museum observatory to the South West Rift zone (**Figure 5**).

FIGURE 3
Artificial tree branches hidden in the Ohia trees.

FIGURE 4
Mounting artificial tree branches in the Ohia trees.

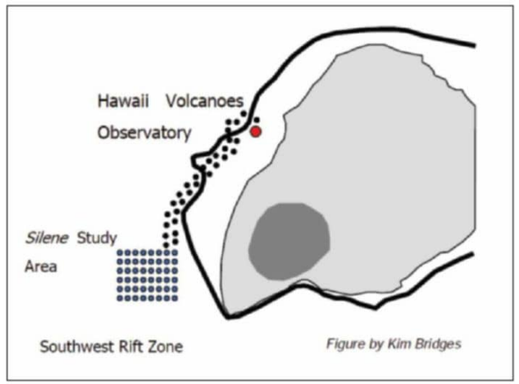


FIGURE 5
Map showing placement of nodes in the Hawaiian Rift Zone.

FEATURES

FIGURE 6
PVC pipes covered with laser prints of Ohia bark.

FIGURE 8
A finished fake rock. Pantone color swatches were used to compare the colors of the "real rocks," for more accurate tinting of the laboratory-made Bondo rocks.

FIGURE 7
Underside of the fake rock molded from real rocks from the area, using tinted Bondo layered into latex molds.

FIGURE 9
Inside of a fake rock containing a microprocessor board. Both the fake rocks and fake branches were RF-invisible, allowing sensors to be hidden in plain sight, while allowing the radios inside them to talk to each other.

The fake tree branches were PVC pipes covered with laser prints of Ohia bark (**Figure 6**), and the rocks were molded from real rocks from the area with tinted Bondo layered into latex molds (**Figure 7**). We used Pantone color swatch books to compare the colors of the "real rocks," so that we could more accurately match the tinting for our laboratory-made Bondo rocks (**Figure 8**). Both Bondo and the PVC pipes were RF-invisible, allowing the team to hide our sensors in plain sight in the rocks (**Figure 9**) and trees, while allowing the radios inside them to talk to each other.

Figure 10 shows a block diagram showing the components of a node in a rock, and **Figure 11** is a photo of the actual microprocessor boards and sensors that were inside each node.

The rocks were realistic enough that I had a standing bet with the USGS folks that I would hide a $50 bill under one of my rocks amidst real rocks. If they could find it without touching, I would give them the $50. In the 5 years of the grant, I never gave away the $50.

Of note—since we were funded by DARPA, we were asked to teach the military how to make those fake rocks.

ZIGBEE AND LORA MESH NETWORKS

Enough history lessons! While quite a few mesh network systems exist, I'm going to talk about two that have become commercially popular and have versions available to the hobbyist: Zigbee and the Z-Wave variation; and LoRa. Common to both Zigbee and LoRA is integral strong encryption. It gets around security issues all too common in inexpensive Wi-Fi Internet of Things implementations that often provide a vulnerability path into your home network.

Zigbee:

Zigbee—a wireless communications protocol—is commonly used in applications with low data rates that require secure networks and long battery life. It is a spinoff from the DARPA SenseIT project, and Andy Wheeler has become a driving force for the Zigbee alliance. The Volcano National Park (PODS) project discussed earlier utilized unlicensed 900MHz radios, but commercial Zigbee is 2.4GHz, instead.

One of the largest Zigbee applications is also the simplest. Water and power utilities use Zigbee from your home meter (water or power) to harvest usage and diagnostic data from meter reader vehicles at the curb, in a simple point-to-point configuration. With only two nodes involved, the mesh network capability isn't used, but encryption is. This eliminates the need for technicians to wander around on private property and run into potentially aggressive "pets" protecting the meter area of the home.

The original intent for Zigbee was to allow sensor data to be relayed out of the massive pipe farms for wastewater processing plants, where the metal pipes would reflect the signals and water would absorb it. The problem for many heavy industrial applications is that data cables all too often get broken or mashed during repairs or maintenance. Having the ability to bucket brigade sensor data reliably became the way that the stench of sewage was avoided.

LoRa, LoRaWAN, and Gateways:

LoRA, short for Long RAnge [2], deviates from Zigbee in that it has been implemented on multiple radio frequencies, depending

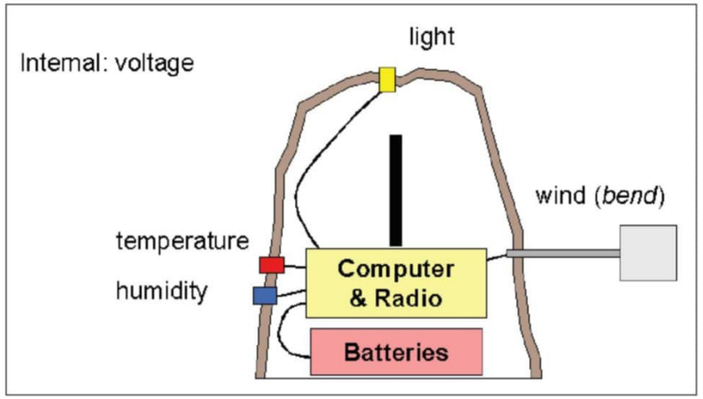
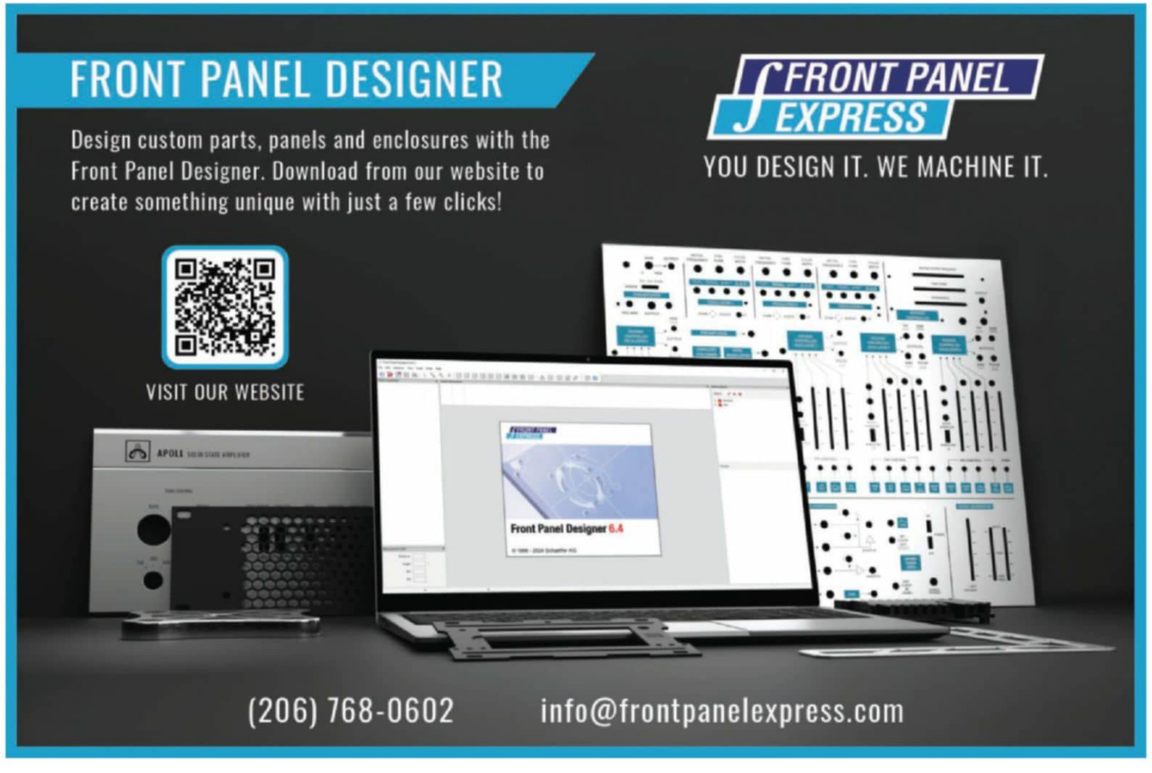


FIGURE 10
Block diagram of a node, showing the microprocessor board and sensors on top of a power source in fake rock.


FRONT PANEL DESIGNER
Design custom parts, panels and enclosures with the Front Panel Designer. Download from our website to create something unique with just a few clicks!
FRONT PANEL EXPRESS
YOU DESIGN IT. WE MACHINE IT.
VISIT OUR WEBSITE
Front Panel Designer 6.4
(206) 768-0602
info@frontpanelexpress.com

FEATURES

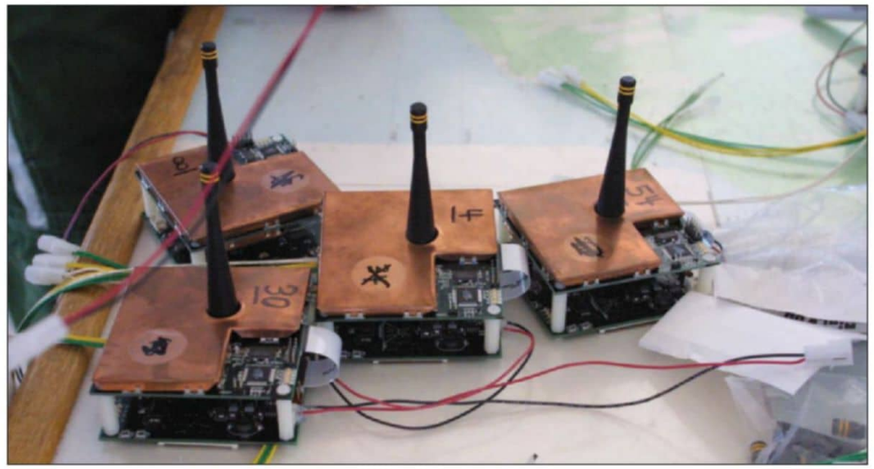


FIGURE 11
Photo of a group of nodes before placement in fake rocks or branches.

upon where in the world you are. It ranges from 433MHz in Asia, to 868MHz in Europe and 915MHz in North America. (Some situational testing by folks in Canada has yielded up to 23km through downtown Toronto). Lower frequencies means lower data rates, but in most cases longer ranges [3].

Of note—the old pager network used VHF radio between 35MHz and 93MHz that allowed pager signals to reach into areas unreachable to mobile data networks. So at 915MHz for LoRA in North America, there exists the potential for to get to areas that cellular data networks have no hope of reaching. My personal testing was able to reach the basement of a shielded research building at the University of Hawaii that has zero cellular coverage.

LoRA, itself, defines the peer-to-peer communications system that includes features such as low power, spread spectrum, and long range—but at the cost of lower data rates. But LoRA developers have gone a bit further and implemented a gateway. They came up with LoRaWAN (a low-power, wide-area networking protocol, designed to link battery-powered devices over long distances), in which the LoRA peers/nodes lean toward the gateway device.

I happen to use the IoThinks system [4] through a MikroTik Knot LR8 LoRA Gateway [5] that can utilize a wired Ethernet or LTE cellular connection to relay LoRA sensor data to the IoThinks cloud servers.

Don't think you have to use a cloud service! Gateways such as the MultiTech Gateways [6] are perfectly happy to send that data to an on-premises server app or many other cloud systems—for example, the AWS IoT Core for LoRaWAN [7]. If you're not inclined to do it yourself, I'm using the YoLink sensor system [8] for my home. It comprises a 1/4-mile effective range for temperature+humidity, gate/door, power control, and valve control, in a turnkey application of the LoRA technology.

From a security standpoint, the gateway implementation also allows me to setup my IoT gateway on an isolated network segment, to provide a security isolation that reduces the potential attack surface for my home or office.

BEFORE YOU TAKE THE PLUNGE...

For those just wanting to get their toes wet, I suggest taking a look at Node-RED, a drag-and-drop, zero-coding solution initially developed by IBM [9]. I made a simple system using the Seeed Studio WIO Node [10] Wi-Fi IoT modules with PIR motion sensors and temperature+humidity sensor that relayed data to Node-RED running on a Raspberry Pi 2 Model B in another building.

Node-RED tallied the motion triggers and pushed hourly summaries, along with hourly temp+humidity readings, into a Google Cloud Spreadsheet. Those numbers helped to justify why my university should keep the air conditioning on for the history department instead of shutting it down as a cost-saving measure.

This project also could have been implemented easily on LoRaWAN, if I hadn't had access to a campus Wi-Fi system. LoRaWAN has become so widespread that a major cable data provider has set up LoRaWAN gateway nodes co-located with their public Wi-Fi Access Points, as an experiment to explore it as an alternative to Zigbee and cellular for city utilities.

Several cities seem to have implemented LoRA-based sound and environmental sensors to measure city-center traffic impact. I'm looking at using LoRa with waterproof temperature sensors to measure overall HVAC health for a local expo center, to help keep the HVAC vendors honest. To be truthful, I could just as easily use Wi-Fi, but the decision was to separate environmental sensors from the expo Wi-FI system to avoid potential loss of connectivity due to expo Wi-Fi congestion. Also, considering that the Wi-Fi is shared with exhibitors, we wanted to separate control systems from public Internet access to reduce our attack surface.

THE ONLY CONSTANT IS CHANGE

The number of tools is continually changing. Sadly, Cisco Systems abandoned its LoRaWAN gateway product as of January 1, 2025, and will fully withdraw from the market by December 31, 2029 [11].

While this saddens me, the IoT world's adoption of LoRa and Zigbee is very healthy worldwide, with companies like Seeed Studio [12]—an avid supporter of Maker Faire in Shenzhen in China—and Adafruit Industries out of New York [13]. Both provide an amazing collection of tutorials on how to use both Zigbee and LoRA IoT RF systems to explore and control the world around us.

If you are inclined to make your own systems, keep in mind that toolkits and libraries for Zigbee and LoRa tend to be specific to the development platform you choose. So, for instance, there's one for the Heltec ESP32 Arduino variation [14], one for the Raspberry Pi [15], and one for Node-RED and MQTT [16].

The key to all my designs is keeping my eye on the goal, while, as much as possible, preventing the unfortunate security vulnerabilities that have given many IoT applications a bad name.

ABOUT THE AUTHOR

Brian Chee is an IT professional with experience spanning over three decades in various roles, including IT Specialist, Systems Programmer, and Computer Scientist. Now on the Board of Directors for Central Florida Fair, Brian has been a Senior Contributing Editor for *InfoWorld* Magazine and host of TWiT LLC's *This Week in Enterprise Technology*. Brian's career includes significant contributions at the University of Hawaii at Manoa, where his expertise in network systems and high-speed networking developed. Brian holds a BS in Computer Science from Chaminade University of Honolulu, and studied Information and Computer Sciences at the University of Hawaii at Manoa.

Additional materials from the author are available at: www.circuitcellar.com/article-materials

References [1] to [16] as marked in the article can be found there.

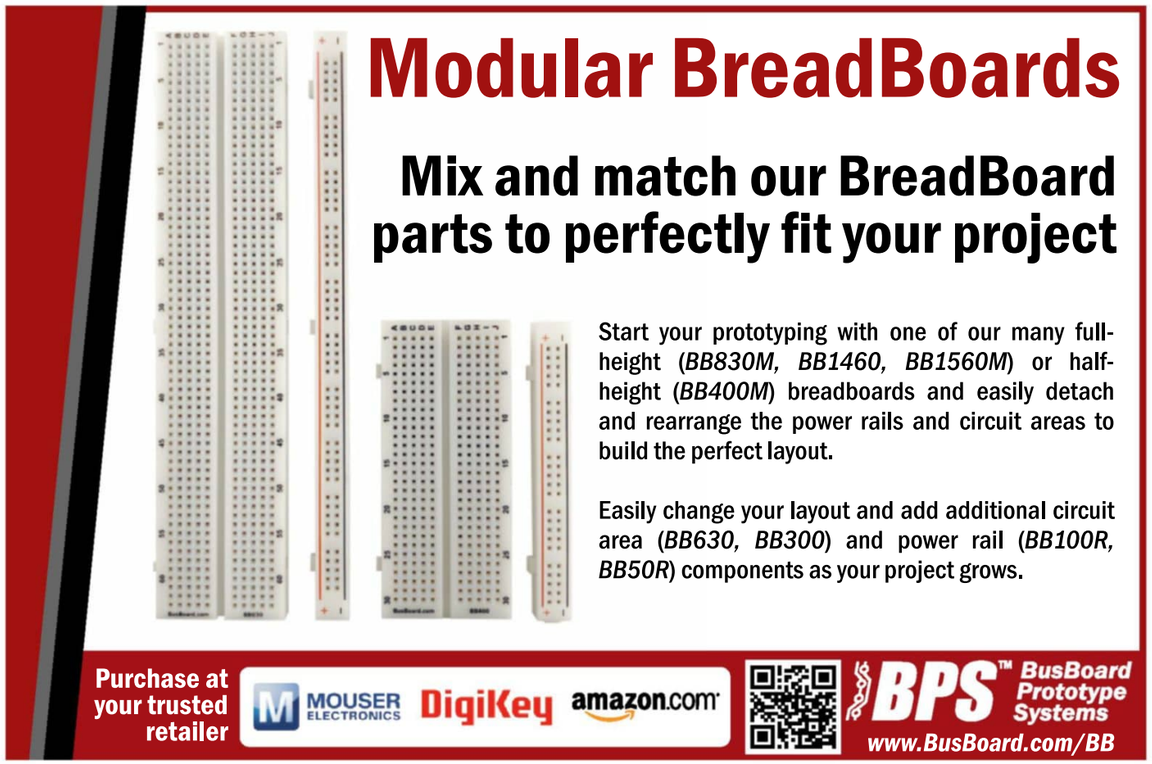
RESOURCES

AdaFruit Industries | www.adafruit.com

IOThink Solutions | iothink-solutions.com

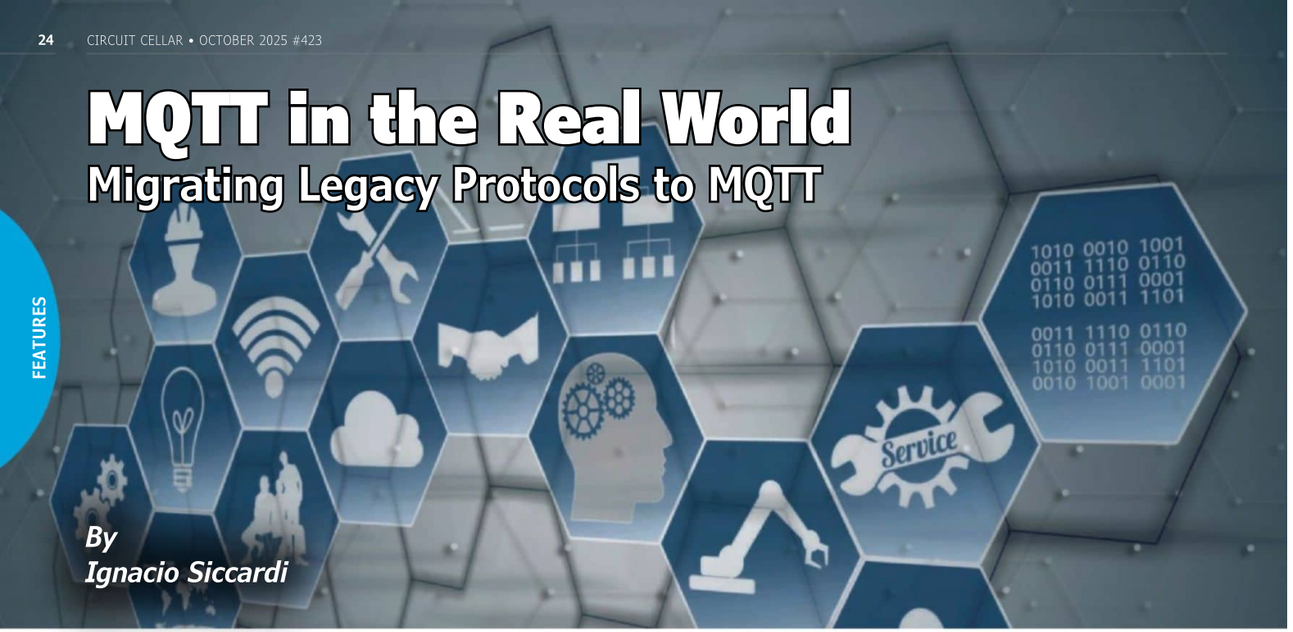
Node-RED | nodered.org

Seeed Studio | www.seeedstudio.com


Modular BreadBoards
Mix and match our BreadBoard parts to perfectly fit your project
Start your prototyping with one of our many full-height (BB830M, BB1460, BB1560M) or half-height (BB400M) breadboards and easily detach and rearrange the power rails and circuit areas to build the perfect layout.
Easily change your layout and add additional circuit area (BB630, BB300) and power rail (BB100R, BB50R) components as your project grows.
Purchase at your trusted retailer
MOUSER ELECTRONICS
DigiKey
amazon.com
BPS™ BusBoard Prototype Systems
www.BusBoard.com/BB

MQTT in the Real World

Migrating Legacy Protocols to MQTT



By Ignacio Siccardi

MQTT is an OASIS standard messaging protocol for the Internet of Things. This lightweight protocol can simplify and improve messages between IoT components that started out using different languages.

Many industrial systems still rely on Modbus, OPC-DA, Profibus, and other legacy fieldbus protocols. While these protocols have served industry well, they present limitations in flexibility, scalability, and cloud integration.

MQTT serves as a bridging solution, enabling seamless migration from legacy systems without requiring complete infrastructure overhauls. Gateways or protocol converters are deployed at the edge to translate legacy data formats into MQTT messages, which are then published to centralized brokers or cloud platforms.

A manufacturing plant uses SCADA with Modbus TCP, integrated with MQTT as a wrapper layer. By employing an edge gateway with Modbus-to-MQTT translation, they maintained existing PLCs while transmitting real-time data to a cloud analytics platform, as shown in **Figure 1**. This enabled the operations team to monitor key parameters remotely while gradually phasing out the legacy system.

The publish-subscribe nature of MQTT enables decoupling between devices and applications, paving the way for sustainable modernization. This transition approach reduces downtime and risks while allowing gradual integration of AI, ML, and advanced analytics.

REAL-TIME DIAGNOSTICS

Traditional diagnostic methods in industrial settings often depend on periodic polling or manual inspection. These approaches are inefficient, particularly when uptime is critical. MQTT provides a lightweight, event-driven alternative that enables continuous data flow for real-time diagnostics.

MQTT is an improvement over previous protocols for a number of reasons. First, it features low overhead and small message size. Next, there are persistent sessions with Quality of Service (QoS) guarantees. Finally, MQTT represents a Last Will and Testament (LWT) for device health monitoring.

A global energy provider installed vibration and temperature sensors on turbine equipment across multiple sites, as shown in **Figure 2**. These sensors sent data via MQTT over cellular and LoRaWAN networks to a centralized diagnostics engine. With QoS level 1, messages were reliably delivered even in low-bandwidth environments.

Machine learning models analyzed incoming data in real-time to identify anomalies like bearing wear or shaft misalignment. When thresholds were exceeded, the system sent alerts to field engineers, reducing downtime by 30% and preventing catastrophic failures. MQTT's support for edge-to-cloud connectivity ensures that diagnostic intelligence can be distributed and responsive to site-specific conditions.

HiveMQ enables real-time diagnostics in smart manufacturing by supporting high-frequency data streaming via MQTT. Manufacturers utilize this system to identify anomalies, continuously monitor equipment, and activate ML models at the edge or in the cloud for predictive maintenance and minimizing downtime.

RISKY REMOTE MAINTENANCE

In sectors such as oil & gas, mining, and maritime logistics, physical access to equipment can be risky, expensive, or logistically complex. Remote maintenance powered by MQTT unlocks a new level of operational efficiency. An offshore drilling platform equipped its pump control systems with MQTT-enabled IoT gateways, as seen in **Figure 3**.

These gateways collected telemetry from legacy programmable logic controllers (PLCs) and exposed command interfaces through MQTT topics. Engineers onshore could subscribe to system metrics and publish configuration updates or firmware patches in response to emerging issues. Crucially, the MQTT broker—hosted in a hybrid edge-cloud setup—enabled asynchronous communication with built-in buffering, ensuring data integrity even during intermittent connectivity.

The result: a 40% reduction in maintenance site visits and improved worker safety. A leading upstream Oil & Gas company uses MQTT to enable secure, remote monitoring and maintenance of oil field equipment. MQTT allows engineers at headquarters to remotely access telemetry, send configuration changes, and initiate operations—all without dispatching field teams, improving safety and reducing operational costs.

RELIABLE OPS DATA

Industrial operations depend on timely and accurate data for process control, quality assurance, and safety compliance. MQTT's lightweight protocol, combined with configurable QoS levels, makes it ideal for transmitting mission-critical data across various industrial topologies.

A food processing facility uses MQTT to transmit data from temperature, humidity, and pressure sensors in real-time. Each sensor publishes to a distinct topic namespace (e.g., /plant1/line3/temp). A centralized broker aggregates data for analytics, regulatory compliance, and alarm systems.

To ensure reliability, the system uses QoS level 2 for critical sensor streams (exactly one delivery) and retains the latest sensor values for late subscribers. Integrations with Grafana and InfluxDB allow visualization and alerting, while AI modules predict potential quality deviations.

The protocol's lightweight footprint minimizes network congestion, even with thousands of concurrent clients. Additionally, MQTT's compatibility with edge brokers such as Eclipse Mosquitto or HiveMQ allows seamless deployment in hybrid IT/OT environments.

Tesla utilizes MQTT extensively within its fleet telemetry system, as seen in **Figure 4**. The company's internal documentation for its Fleet Telemetry datastore indicates that vehicle data, alerts, errors, and connectivity status are published to an MQTT broker. This architecture supports high-frequency, mission-critical operational data delivery, enabling real-time monitoring and alerting capabilities across a global fleet.

The MQTT setup ensures reliable publish-subscribe messaging from hundreds of

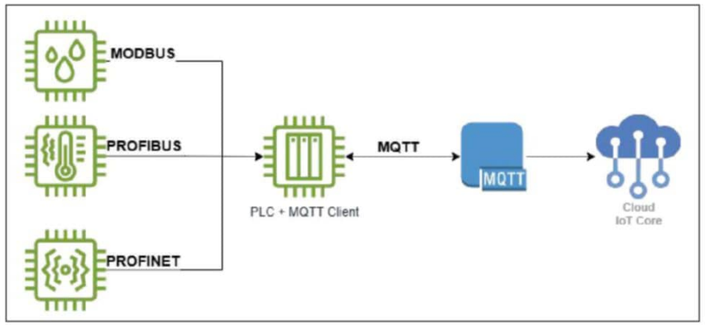


FIGURE 1
MQTT can be implemented as a wrapper that allows incumbent sensors or edge devices to use the modern protocol.

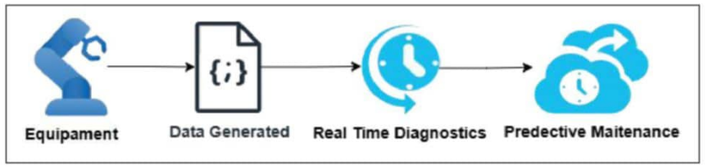


FIGURE 2
MQTT is frequently used to deliver sensor data on critical infrastructure, even in the face of unreliable communications.

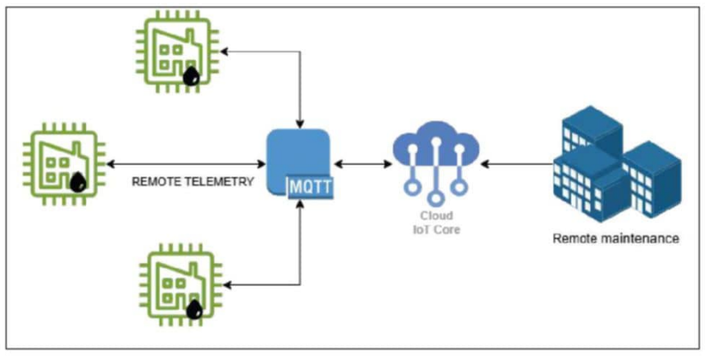


FIGURE 3
MQTT implemented as a gateway provides efficient communications from a fleet of devices in a central locations back to a central management hub.

ABOUT THE AUTHOR

Ignacio Siccardi is an Information Systems Engineer specializing in IT infrastructure and IoT. He currently works on the deployment of IoT infrastructure with LoRaWAN in Argentine agro-industries.

You can learn more about Ignacio in his personal blog www.rosariot.com.

FEATURES

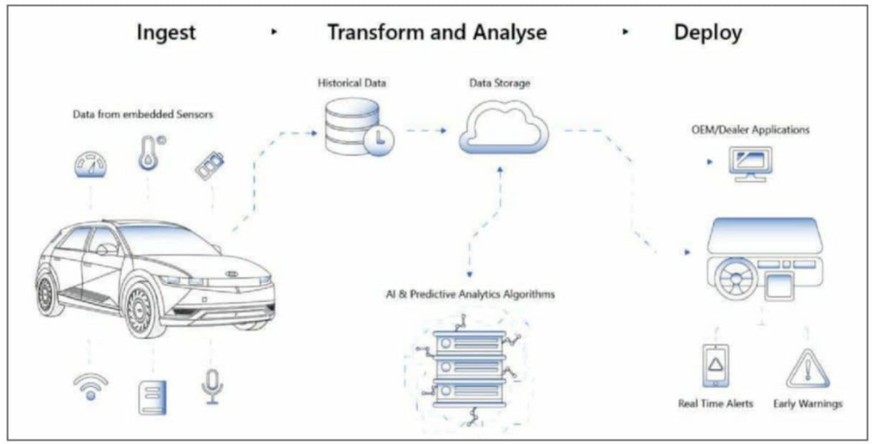


FIGURE 4
Tesla is an example of an auto manufacturer using MQTT for communications from a mobile platform subject to inconsistent communications quality.

thousands of vehicles, capturing telemetry data such as battery status, error codes, GPS position, and connectivity metrics.

These MQTT messages are the backbone of Tesla's remote diagnostics and operational dashboards that field technicians and engineers use to detect critical issues in near real time.

ONE-WIRE SENSOR SETUP

The DS18B20 digital temperature sensor, though small and affordable, is available in industrial-grade, sealed stainless-steel cases with ingress protection ratings (IP65–IP68). These ruggedized versions make the DS18B20 suitable for demanding environments such as food processing plants, HVAC control systems, manufacturing lines, and remote environmental monitoring stations.

DS18B20 offers distinct advantages for industrial applications:

- Digital Output over 1-Wire Bus: This eliminates the need for analog signal conditioning, reducing the system's susceptibility to electrical interference in electrically noisy industrial environments.
- High Measurement Precision: The sensor provides ±0.5°C accuracy over the critical –10°C to +85°C range, with a resolution programmable from 9 to 12 bits.
- Rugged Mechanical Form Factors: Probes are available with stainless-steel housings, epoxy sealing, and extended temperature ranges up to +125°C.
- Unique 64-bit Serial Codes: These enable multiple sensors to coexist on the same 1-Wire bus without collision, simplifying wiring in distributed sensing topologies.

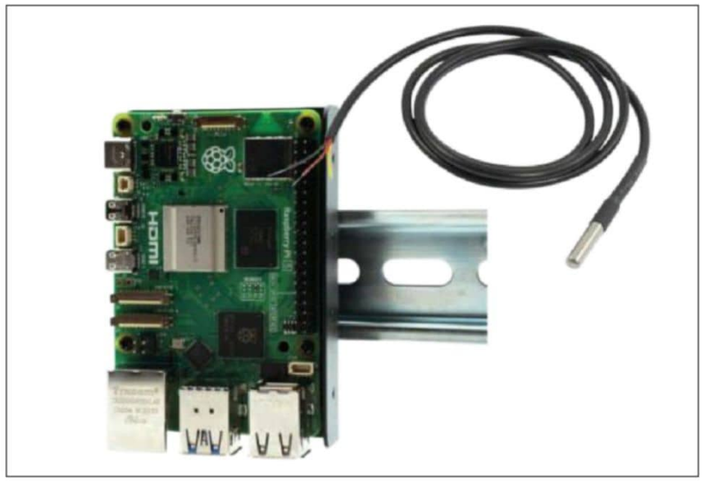


FIGURE 5
The Raspberry Pi, an incredibly common platform in embedded systems, provides plenty of compute horsepower and usable I/O for MQTT purposes.

The Raspberry Pi, shown in **Figure 5**, offers an excellent edge computing platform for integrating industrial DS18B20 sensors into an MQTT-based system. It combines low-level GPIO access with the capability to run full IoT stacks, data analytics frameworks, and secure communication protocols.

STANDARD MESSAGING

MQTT is an industrial standard that doesn't need a huge enterprise for support. A lightweight, simple-to-implement protocol, MQTT can make life easier for developers and administrators while keeping deployed projects on the same page as hundreds of supporting infrastructure products.

Additional materials from the author are available at: www.circuitcellar.com/article-materials

RESOURCES

MQTT | www.mqtt.org

INSTALLATION GUIDE

Enable 1Wire on Raspberry Pi
sudo raspi-config and select "Interfacing Options".
Then select: "P7 1-Wire" as seen in **Figure 1**.
After reboot:
lsmod | egrep 'w1_gpio|w1_therm' # should list both
ls /sys/bus/w1/devices/
should see 28-XXXXXXXXXXXX for DS18B20

Raspberry Pi setup (GPIO, 1-Wire enablement)
Connect the data pin of the DS18B20 to GPIO 4 (BCM 4)
Connect the VCC pin of the DS18B20 to the 3.3 V pin
Connect the GND pin of the DS18B20 to a GND pin
Ensure that a 4.7 kΩ pull-up resistor is connected between the data pin (DS18B20) and the VCC pin (3.3 V) of the sensor as diagrammed in **Figure 2**.

Install dependencies
sudo apt update
sudo apt install -y python3-pip
pip3 install w1thermsensor paho-mqtt

Choose a unique MQTT topic
Example broker:
iot/circuit_cellar/lab/rpi/temperature

Publisher script (Python)
Saves as ds18b20_to_hivemq.py:

```
#!/usr/bin/env python3
import json, time, socket, os, sys
from datetime import datetime, timezone
from w1thermsensor import W1ThermSensor, Unit
import paho.mqtt.client as mqtt
BROKER = "broker.hivemq.com"
PORT   = 1883
ORG    = "circuit_cellar"          # <<< change
DEVICE = "rpi"         # <<< change
TOPIC_BASE = f"iot/{ORG}/lab/{DEVICE}/temperature
TOPIC_DATA = TOPIC_BASE
TOPIC_STATUS = f"{TOPIC_BASE}/status"
TOPIC_META = f"{TOPIC_BASE}/meta"
CLIENT_ID = "LAB-CIRCUIT-CELLAR"
# --- MQTT setup ---
cli = mqtt.Client(mqtt.CallbackAPIVersion.VERSION2, client_id=CLIENT_ID, clean_session=True)
cli.will_set(TOPIC_STATUS, payload="offline", qos=1, retain=True)
def on_connect(client, userdata, flags, reason_code, properties=None):
    print(f"[MQTT] Connected: rc={reason_code}")
    # Announce online
    client.publish(TOPIC_STATUS, "online", qos=1, retain=True)
    # Publish metadata once
    meta = {
        "sensor": "DS18B20",
        "unit": "°C",
        "qos": 1,
        "retain": True,
        "host": socket.gethostname(),
        "topic": TOPIC_DATA
    }
    client.publish(TOPIC_META, json.dumps(meta), qos=1, retain=True)
def on_disconnect(client, userdata, reason_code, properties=None):
    print(f"[MQTT] Disconnected: rc={reason_code}")
cli.on_connect = on_connect
cli.on_disconnect = on_disconnect
# Connect (public, no auth)
```

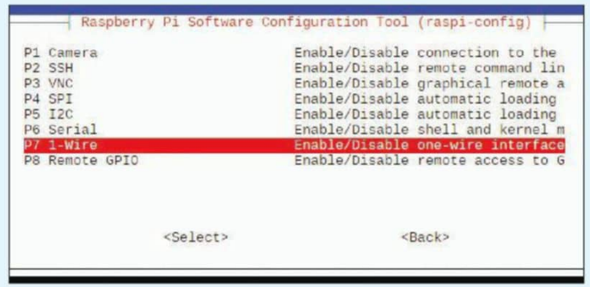

Raspberry Pi Software Configuration Tool (raspi-config)
P1 Camera Enable/Disable connection to the
P2 SSH Enable/Disable remote command lin
P3 VNC Enable/Disable graphical remote a
P4 SPI Enable/Disable automatic loading
P5 I2C Enable/Disable automatic loading
P6 Serial Enable/Disable shell and kernel m
P7 1-Wire Enable/Disable one-wire interface
P8 Remote GPIO Enable/Disable remote access to G
<Select>
<Back>

FIGURE 1
Raspberry Pi Configuration Tool - Enable 1-Wire

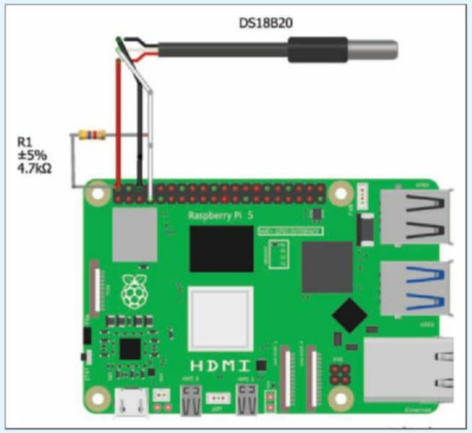


FIGURE 2
Temperature probe connected to Raspberry PI 5 with pull-up resistor

```
cli.connect(BROKER, PORT, keepalive=60)
cli.loop_start()
# --- Sensor setup ---
try:
    sensor = W1ThermSensor()  # auto-detect first DS18B20
except Exception as e:
    print(f"[ERROR] DS18B20 not found: {e}")
    sys.exit(1)
def read_celsius():
    # Tolerant read with simple retry
    for _ in range(3):
        try:
            return round(sensor.get_temperature(Unit.DEGREES_C), 2)
        except Exception:
            time.sleep(0.5)
    raise RuntimeError("Failed to read DS18B20 after retries")
# --- Main loop ---
INTERVAL_S = 5
while True:
    try:
        value = read_celsius()
        payload = {
            "ts": datetime.now(timezone.utc).isoformat(),
            "value": value,
            "unit": "°C",
            "sensor": "DS18B20",
            "device": DEVICE
        }
        # QoS 1 + retain keeps the latest value available to late subscribers
        info = cli.publish(TOPIC_DATA, json.dumps(payload), qos=1, retain=True)
        info.wait_for_publish(timeout=5)
        print(f"[PUB] {TOPIC_DATA} -> {payload}")
    except Exception as e:
        print(f"[ERROR] {e}")
    time.sleep(INTERVAL_S)
```

Run it:
python3 ds18b20_to_hivemq.py

Subscribe to verify (from any machine)
See live values:
mosquitto_sub -h broker.hivemq.com -t 'iot/circuit_cellar/lab/rpi/temperature' -v
Optional: run as a service
sudo tee /etc/systemd/system/ds18b20-mqtt.service >/dev/null <<'UNIT'
[Unit]
Description=DS18B20 to HiveMQ MQTT Publisher
After=network-online.target
Wants=network-online.target
[Service]
Type=simple
User=pi
WorkingDirectory=/home/pi
ExecStart=/usr/bin/python3 /home/pi/ds18b20_to_hivemq.py
Restart=always
RestartSec=5
[Install]
WantedBy=multi-user.target
UNIT
sudo systemctl daemon-reload
sudo systemctl enable --now ds18b20-mqtt

The Premier Online Conference for Embedded Systems Engineers
Advance Your Skills. Accelerate Your Career.
Expert-led training at an unbeatable value
Learn on your schedule—unlimited on-demand access until Dec 2025
Actionable insights you can apply immediately
Don't fall behind—stay ahead with the latest in embedded systems.
Embedded Online Conference
KEYNOTE SPEAKERS
Jacob Beningo
Elecia White
Philip Koopman
James Grenning
Register now and take your expertise to the next level!
Learn More: →
www.embeddedonlineconference.com

TECH FEATURE

Embedded AI

AI Comes to the Operational Network's Edge

By
Curtis Franklin,
Editor-In-Chief

New developments in single-board computers and the chips that power them are bringing the dream of operational artificial intelligence in devices at the network's edge closer to reality than ever before. We're looking at how close we are to widely distributed, on-device intelligence and which trends and developments seem most likely to push us across the goal line.

The integration of Artificial Intelligence (AI) into Internet of Things (IoT) edge devices marks a fundamental shift from centralized cloud computing to distributed intelligence, bringing AI processing closer to data as it's generated and used. This shift, known broadly as Edge AI, is rapidly transforming various industries by enabling devices to process data and make control decisions locally without relying on distant cloud servers or models controlled by third parties.

A SMARTER EDGE

Edge AI is deploying artificial intelligence models directly on edge devices, including sensors, microcontrollers (MCUs), or gateways, enabling them to process data locally. This local processing capability is crucial for enhancing the functionality and efficiency of embedded systems.

The core concepts of Edge AI center on bringing computational power to the network's periphery. This approach offers several significant advantages over traditional client-server or cloud computing. Many of the advantages build on the basic benefits of not shipping data back and forth over long distances, but those benefits show themselves in a number of different ways.

The most visible (and currently, the most consequential) benefit of AI at the edge is in real-time decision-making. Edge AI is crucial for autonomous systems, such as drones, industrial robots, and autonomous vehicles, where rapid decision-making is essential. Control loops in embedded process control systems often require responses within milliseconds, making deterministic execution essential. By removing the need to send data to the cloud and wait for a response, Edge AI greatly lowers latency, often providing responses in milliseconds.

Keeping data and the AI model local also contributes to enhanced privacy and security. Sensitive data stays on the device, reducing the risk of third-party breaches and improving privacy. By eliminating constant data communications, edge AI eliminates a significant attack vector for threat actors. Local, edge AI allows secure, offline operation, the sort of operational mode that is especially important in regulated settings and for safeguarding intellectual property.

Local data and model deployment has the happy side effect of requiring less bandwidth than the distributed model needs. Only relevant data, usually the inference result instead of raw sensor data, is transmitted. And transmitting the result rather than the raw data will also require less storage capacity than is needed or massive sensor dumps. Taken together this significantly reduces bandwidth and storage consumption and helps alleviate network congestion and budgetary concerns.

Although there is a one-time hardware investment, the ongoing cost for Edge AI is usually low, unlike the pay-as-you-go model of cloud inference calls. For the engineers who also think about total cost of ownership, this slides the vast majority of the cost into the CAPEX (capital expenses) budget rather than the OPEX (operating expenses). There are tax and profit/loss implications, but those are topics for another magazine.

While it is being financially responsible, edge AI can also be more reliable than widely connected systems that require constant network communications. Edge devices can function independently of network connection, allowing them to remain operational during outages. This is especially important in remote industrial and environmental monitoring sites.

NOT A CLOUD

Understanding edge AI and its potential advantages requires understanding just how it's different than traditional cloud-based AI, particularly large language models (LLMs). It starts with the fact that they serve different purposes and operate under different constraints.

The first, and most obvious, difference is in the compute location. Edge AI runs directly on devices such as microcontrollers, gateways, or AI accelerators, with inference happening locally. Cloud-based LLM AI, conversely, executes on powerful remote servers or GPU clusters such as those shown in **Figure 1**, requiring internet connectivity to access models via APIs. One of the reasons that the locations are different is that there are vast differences in the size and complexity of the model, as shown in **Figure 2**.

Edge AI utilizes compact, quantized, or pruned models, typically with millions to tens of millions of parameters, trading off some accuracy for efficiency and speed. Cloud-based LLM AI leverages massive models, often with hundreds of millions to hundreds of billions of parameters, achieving state-of-the-art language understanding and generation.

The differences in size and complexity speak to the differences in mission: Edge AI is highly targeted, focused on responding to the needs of the immediate task at hand while cloud-based LLM tends to be more generalized, trying to answer any question on any topic that might be presented. The differences in mission mean that each can benefit from the latency involved in their communications.

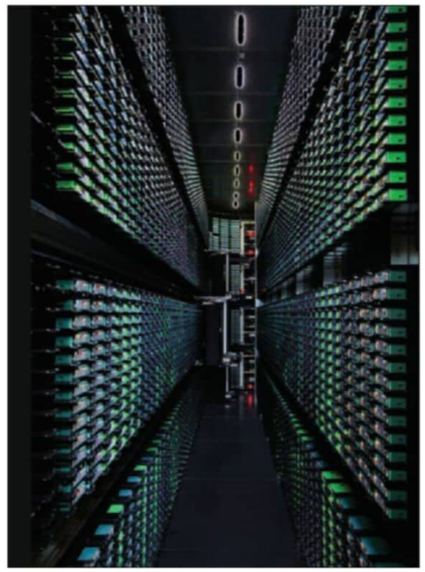


FIGURE 1
In 2025, the average AI cloud-scale data center is up to 2 million sqaure feet of space while consuming between 200 megawatts and 2 gigawatts of power. That marks a power consumption per square foot roughly three times that of a legacy data center.

Edge AI offers millisecond-scale latency as data never leaves the device, which also significantly reduces bandwidth usage. Cloud-based LLM AI incurs round-trip delays over networks, typically tens to hundreds of milliseconds, and consumes more bandwidth by sending data back and forth. On the other hand, edge AI is often responding to real-world, real-time conditions that require immediate response while cloud-based LLMs are most often asked questions for which the processing time will be far more significant than the communications latency.

The amount of processing required is also driving the difference in energy usage for each. Edge AI is optimized for low energy budgets and operates with limited CPU/accelerator cores and small RAM. It will typically operate within the 5V to 48V power limitations of most non-AI edge computing devices. Cloud-based LLM AI, on the other hand, benefits from high power draw in data center GPUs and virtually unlimited compute and memory. Cloud-based LLM AI is the type of AI driving conversations

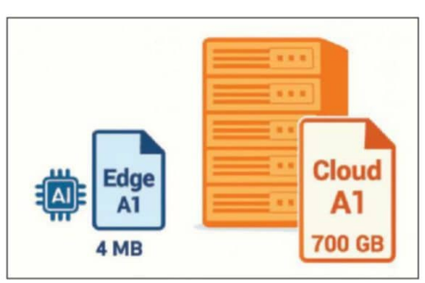


FIGURE 2
The difference in model size reflects the different missions of edge and cloud AI.

TECH FEATURE

around the impact that AI is having on power grids and global energy demand.

TECHNOLOGY FOR EDGE AI

Deploying AI on resource-constrained edge devices is possible because of a combination of specialized hardware, sophisticated software, and optimized model architectures. Edge AI hardware integrates general-purpose computing with specialized accelerators to perform inference under tight power, latency, and cost constraints. Within the category there are multiple building blocks from both the hardware and software sides of the component fence. While much has been written about the software engines used for AI, without the proper hardware, inferences cannot be performed in time to make useful embedded system decisions.

EDGE AI HARDWARE

Microcontrollers and microprocessors are the core processors for control and computation in the vast majority of embedded edge systems, with examples such as the ARM Cortex-M and RISC-V, along with specialized microcontrollers, like such as ARM Cortex-M series, incorporating neural processing units that are becoming increasingly common for embedded AI. Depending on the specifics of the use case, engineers can now choose between microcontrollers paired with outboard AI accelerators or AI-ready boards that have both microprocessors and AI accelerators together with the glue circuitry required for their happy cooperation.

AI accelerators are designed to speed up inference tasks, providing high efficiency and higher speeds for AI workloads. When looking at the specifications for AI accelerators of any type, there are terms you should understand.

First, TOPS in AI performance stands for Trillions (or Tera) Operations Per Second, and it is a metric used to describe the computational speed of AI hardware such as Neural Processing Units (NPUs), Graphics Processing Units (GPUs), or other AI accelerators. Next, you may see performance expressed in terms particular to a specific model. "Capable of running Llama2-7B at 10 tokens/s under 5W," for example, means that a device can execute the Llama2-7B language model—which has 7 billion parameters—producing 10 tokens of output per second while consuming less than 5W of power. In this context, "tokens" are words (or meaningful word fragments) provided as output.

While a growing list of edge AI accelerators is available for engineers to use, a handful command a significant market share and provide a useful snapshot of capabilities.

- Google Edge TPU: A dedicated ASIC for efficient inference at the edge. The Google Coral Dev Board, for example, pairs an NXP i.MX 8M SoC with an Edge TPU, offering 4 TOPS at ~2W for smart cameras.
- Intel x7000RE: A deep-learning VPU with CNN accelerators, widely used in drones and vision modules.
- NVIDIA Jetson Series: Robust System-on-Chip (SoC) platforms like the Jetson AGX Orin (shown in **Figure 3**) integrate multi-core ARM CPUs with powerful GPUs (e.g., Ampere GPU with 2048 cores) and Tensor Cores, delivering up to 275 TOPS for complex multi-sensor AI workloads at configurable power levels (15-60W). These are ideal for autonomous robots and 3D perception.
- Hailo-10: A purpose-built ASIC offering 40 TOPS at INT4, capable of running Llama2-7B at 10 tokens/s under 5W and Stable Diffusion 2.1 in less than 5 seconds per image.
- Kinara Ara-2: Accelerates up to 30 billion INT4 parameters within a 6W envelope, targeting edge servers and laptops.
- Qualcomm Robotics RB5: Features an Octa-core Kryo CPU, Adreno GPU, and Hexagon NPU, delivering ~5 TOPS at ~10W for 5G-enabled robotics and drones.

Modern edge AI hardware often uses heterogeneous SoCs that combine CPUs, GPUs, and digital signal processors. Modular design with chiplet architectures enables the independent development of compute, memory, and I/O blocks, which are then stitched together, driving down costs and shortening the time-to-market for custom solutions.

FIGURE 3
NVIDIA's Jetson AGX Orin (shown with a development kit) uses technology related to that used in cloud AI data centers packaged in a form suitable for edge AI.

Techniques such as fan-out wafer-level packaging and 3D stacking are employed to increase compute density in smaller footprints, thereby reducing interconnect distances and enhancing energy efficiency for devices including drones and wearables. Taken together, these developments are putting more capable AI processing into smaller, more energy-efficient packages.

Those smaller packages don't just contain processors and ASICs, of course. There are other critical components that must be present in order to create useful edge AI. For example, memory components such as MRAM, FeRAM, and EEPROM store models and log data.

MRAM is particularly promising for AI-enhanced edge systems due to its speed, endurance, and low power consumption. As model sizes increase, memory bandwidth becomes a critical bottleneck, leading manufacturers to embed high-bandwidth on-chip SRAM and explore nonvolatile solutions such as MRAM and RRAM to minimize power-hungry DRAM accesses.

Because edge AI is frequently used in mobile applications or facilities that don't allow cabled communications, technologies such as Wi-Fi, Bluetooth Low Energy (BLE), LoRa, and 5G enable data transmission and Over-the-Air (OTA) updates, crucial for maintaining and evolving edge AI systems. Integration with 5G and wireless power (e.g. Powercast's RF tech) is removing barriers to scale. Additional scale barriers are falling on the software side of the AI equation.

So what level of performance is reasonable for engineers to expect from an edge AI implementation? Edge AI systems typically achieve performance on the level of:

- Latency: 1ms to 50ms (inference on device).
- Throughput: Tens to hundreds of inferences per second.
- Memory footprint: Less than 50MB.
- Power consumption: Milliwatts to a few watts.
- Connectivity dependency: None or intermittent.

These metrics indicate that edge devices can deliver real-time responses essential for robotics, safety systems, or augmented reality. One question for the future is just how far beyond that performance level edge AI systems might ultimately perform.

A MODEL ON THE EDGE

There are a number of different inference engines and software runtimes that can be used on various pieces of hardware to implement AI at the network's edge. These software tools enable the execution of optimized AI models on edge devices. Among the most commonly used are:

- TensorFlow Lite: Uses FlatBuffers for sub-MB model sizes, optimized for ARM and DSPs, and is widely used for full integer quantization workflows. TensorFlow Lite Micro is specifically designed for microcontrollers.
- ONNX Runtime: Supports cross-vendor hardware with graph optimizations and custom kernels.
- PyTorch Mobile: Compiles TorchScript models with ahead-of-time (AOT) optimizations.
- ARM CMSIS-NN: Provides highly efficient neural-network kernels specifically for Cortex-M MCUs.
- Apache TVM: Runs models through an auto-tuning stack to generate hardware-specific code, aiding in hardware-aware optimizations.
- Vendor SDKs: Examples include NVIDIA TensorRT and Intel OpenVINO, which leverage specialized accelerators to boost throughput.
- TinyML frameworks: Specialized for running models on microcontrollers with minimal RAM.

BUILDING A BETTER MODEL

The software ecosystem for Edge AI focuses on optimizing models and providing robust deployment and management tools. Edge AI models are specifically optimized using techniques such as quantization and pruning to fit the low-power, resource-constrained environment of edge devices. Architectures

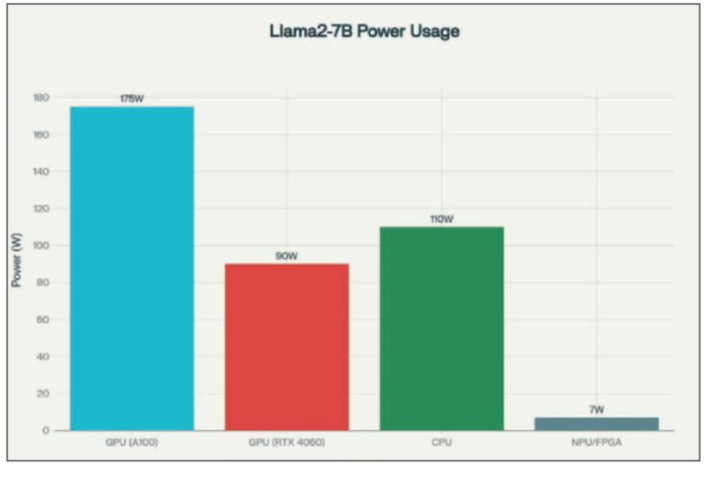


FIGURE 4
The Llama2 AI engine can consume vastly different levels of power depending on where and how it is deployed.

TECH FEATURE

that include MobileNet, Tiny-YOLO, and microcontroller-friendly convolutional neural networks (CNNs) are common. In addition, compression techniques are often used to make the models even more compact, with results such as those shown in **Figure 4**.

Quantization is a compression technique that reduces 32-bit floating-point numbers to lower-bit representations, such as 8-bit integers or even sub-byte, to significantly shrink the model's memory footprint and speed up integer-only hardware operations. Pruning, on the other hand, involves removing redundant neurons or channels from a neural network to reduce its size and computational load without significant loss in accuracy. And there are more.

Knowledge distillation is a technique where a smaller "student" model is trained to mimic the outputs of a larger, more complex "teacher" model, resulting in a more compact and efficient student model. Still more compact models can be generated by AutoML and Neural Architecture Search (NAS), techniques that automate the generation of bespoke lightweight architectures tailored for specific hardware targets and tasks. Not all architectures require a specific hardware platform for their host, though.

Open formats such as ONNX and Multi-Level Intermediate Representation (MLIR) coupled with compiler toolchains such as Apache TVM and TensorFlow Lite Micro provide hardware-aware optimizations and improve model portability across diverse hardware targets. They allow software developers to write code and build models independent of the hardware decisions that engineers might make.

EDGE AI IN THE REAL WORLD

Edge AI is already making a tangible impact across a diverse range of sectors. In manufacturing, edge AI systems are deployed to detect defects using computer vision, monitor vibration patterns in motors and pumps for predictive maintenance, analyze sensor data for real-time quality control, and optimize workflows.

In traditional industrial settings, AI-enhanced systems can adapt control parameters dynamically based on historical performance, predict equipment failures, and optimize processes. Examples include adjusting temperatures and pressures in chemical processing plants, performing load balancing and fault detection for power grids, and identifying safety hazards in oil and gas operations.

When deployed in smart buildings, edge AI optimizes heating, ventilation, and air conditioning (HVAC) and lighting based on occupancy data, enhancing energy efficiency and improving occupant comfort and safety.

And in healthcare wearable sensors use edge AI to continuously monitor patients and detect emergencies, keeping sensitive data on-device. The continuous monitoring can, after inference is performed, be shared with larger diagnostic AI to better predict patient outcomes and optimize healthcare routines.

EDGE AI INFERS A FUTURE

The field of edge AI is rapidly evolving, driven by ongoing advancements in hardware, software, and AI models. The market is projected to experience substantial growth, from $14.8 billion in 2023 to $163 billion by 2033. Several key trends indicate the trajectory of Edge AI in the near future.

By 2030, intelligence is expected to be embedded in every sensor, actuator, and autonomous system. Devices will no longer passively wait for cloud responses but will sense, reason, and act locally in real-time. This shift will enable factories, vehicles, and homes to operate with self-contained AI that continuously adapts to its environment.

While this is taking place, edge nodes will increasingly train and refine models collaboratively without sharing raw data. Federated learning frameworks will empower swarms of devices—such as autonomous cars, medical wearables, and industrial robots—to improve a shared global model while preserving data privacy. This approach is projected to capture nearly $300 million in market value by 2030.

The edge will also begin to host more comprehensive AI models suitable for broader purposes. Generative AI capabilities, traditionally confined to the cloud, will migrate to purpose-built ASICs at the edge, often contained on hybrid SOCs such as the one illustrated in **Figure 5**.

Chips such as Hailo-10 already demonstrate the ability to run LLMs such as Llama2-7B under 5W, producing text and even multimodal outputs directly on the device. Concurrently, tiny transformer variants are being developed to bring LLM-style reasoning capabilities closer to edge devices. This will unlock on-device assistants, context-aware interfaces, and real-time content creation without network dependency.

The future will see neuromorphic chips, inspired by the human brain's spiking networks, delivering ultra-low-power learning and inference. Additionally, early quantum-enhanced coprocessors will accelerate specific linear algebra kernels for optimization and sampling tasks, redefining efficiency and opening new applications in control, vision, and signal processing.

Multi-agent reinforcement learning setups will allow fleets of edge devices to experiment, learn, and share strategies in real time. This means industrial robots will autonomously detect drift, correct faults, and optimize workflows, while smart infrastructure will dynamically adjust to traffic patterns, energy demand, and environmental conditions with minimal human oversight.

Common Intermediate Representations (IRs) such as ONNX and MLIR will become ubiquitous, enabling seamless model portability across diverse hardware targets. Toolchains will automatically tailor models—quantizing, pruning, and fusing operations—to maximize throughput and power efficiency, lowering the barrier for developers to deploy Edge AI at scale. And while edge capabilities expand, cloud LLMs will continue scaling and integrating edge-feasible distillations to offer the best of both worlds, with a seamless interplay between local and remote AI.

Finally, all of this will be done in a way that is more secure than edge computing today. While current edge deployments are more secure than distributed AI because data does not have to be transmitted at the same level, future edge AI designs will embed root-of-trust modules and post-quantum cryptographic accelerators directly in silicon.

These improvements will secure both model integrity and user data against current tampering methods and future quantum attacks, making edge deployments viable for highly regulated sectors such as healthcare, finance, and critical infrastructure.

THE INTELLIGENT EDGE

AI in IoT edge devices is moving toward an era of ubiquitous, autonomous, and highly efficient intelligence. This shift, driven by continuous innovation in specialized hardware, advanced model optimization techniques, and sophisticated software toolchains, promises to deliver rapidly improving real-time capabilities, enhanced privacy, and robust offline operation across a vast array of applications. It is likely that the improved performance and increased perceived security will enourage reluctant consumers to embrace edge AI more fully on devices they encounter on a daily basis.

The near future will see a proliferation of smart devices capable of complex reasoning, collaborative learning, and even generative AI, with the potential to fundamentally transform how we interact with technology and our environment. Developers are increasingly focusing on lightweight, secure, and scalable applications for edge devices, ensuring the widespread and reliable deployment of this powerful technology.

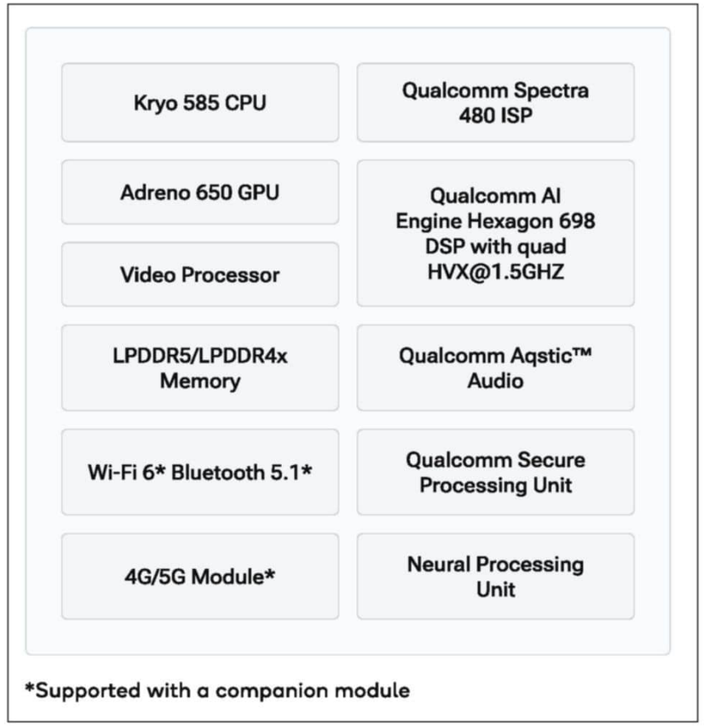


FIGURE 5
Hybrid architecture, containing both CPUs and dedicated AI processors, will be increasingly common as greater inference capability is brought to the edge.

Additional materials from the author are available at: www.circuitcellar.com/article-materials

RESOURCES

Apache | https://apache.org
ARM | https://arm.com
Coral | https://coral.ai
Hailo | https://hailo.ai
Intel | https://www.intel.com
Keras | https://keras.io
Kinara | https://kinara.ai
LLAMA | https://www.llama.com
Luxonis | https://luxonis.com
MLIR | https://mlir.llvm.org
NVIDIA | https://nvidia.com
ONNX Runtime | https://onnxruntime.ai
Powercast | https://www.powercastco.com
PyTorch | https://pytorch.org/
Qualcomm | https://www.qualcomm.com
TensorFlow | https://www.tensorflow.org
TinyML | https://tinymlbook.com

DATASHEET

Datasheet: MEMS Sensors

MEMS Chips Bring Programs and Action Together

By
Curtis Franklin,
Editor-In-Chief

For some applications there are clear divides between the digital and physical worlds. For others, an interface between the two worlds that fits easily into the smallest package is the only solution that makes sense. Enter the Microelectromechanical Systems (MEMS) chip. These powerful chips can enable a host of practical solutions, but the sheer variety of options makes choosing the right chip for the job a critical decision.

Mashups are all the rage. Whether in cuisine, fashion, or music, taking two things that shouldn't work together and slamming them into peaceful coexistence is very, very popular. In the realm of the integrated circuit, MEMS chips are mashups that move edge computing from the purely analytical to the physically meaningful. And the continuing growth of computing in a wide variety of IoT and OT applications means that MEMS chips will only get more important.

MEMS are miniaturized devices that integrate mechanical elements, sensors, actuators, and electronics on a common silicon substrate. Because there are so many possibilities for what a MEMS can do, there are many different individual capabilities available on MEMS. In general, though, a MEMS will feature a single capability and be aimed toward a single use case or industry. This is a class of a device that is very broad in concept and very specific in individual implementation.

SENSORS OF EVERY TYPE

MEMS sensors are tiny microelectromechanical devices that act as sensors; they can detect mechanical, magnetic, positional, or chemical changes (among others) and convert them into electrical information. Depending on their construction, MEMS sensors can measure pressure, motion, acceleration, temperature, or chemical presence and enable important functions in consumer electronics devices, including critical features such as image stabilization or screen orientation in smartphones, position and fitness tracking in wearables, and air quality monitoring in smart home applications.

MEMS sensors are a mashup of electronic components and mechanical hardware. In general, a MEMS sensor contains a MEMS element packed in a semiconductor housing together with an evaluation circuit—the application-specific integrated circuit (ASIC).

WHY MEMS?

Almost everything done by a MEMS element can be done by a series of discrete components. So why choose a MEMS chip? Physical size, power consumption, and circuit simplicity all argue in favor of the MEMS route. While not all are tiny, some MEMS chips are small SMD components, while others are the size of simple ICs easily mounted to a PC board.

MEMS sensors are used in a wide variety of industries and applications. In consumer electronic devices, MEMS sensors are implemented in smartphones, tablets, wearables, hearables, laptops, drones, robots, smart homes, as well as in gaming and AR/VR applications. Smart watches can identify various activities such as standing still, walking, running, cycling, swimming, and numerous other actions. Integrated in hearables, MEMS sensors play an important role in enabling seamless device control through detecting touch, tap, and gestural actions on the device.

MEMS devices have revolutionized the human/machine interface at the network's edge. Here, we present some of the most popular MEMS chips in the world as a guide to the sort of capabilities that are available on devices of all sizes and types.

NEXT MONTH'S TOPIC: DC-DC Converters Send related product announcements to editor@circuitcellar.com

Low-Power Listening

The Analog Devices ADXL1005 delivers ultra-low noise density over an extended frequency range and is optimized for bearing fault detection and diagnostics. The capabilites of the ADCL1005 make it suitable for applications in which it listens for out-of-spec vibrations or component failure indications.

With low power and single-supply operation of 3V to 5.25V, the ADXL1005 also enables wireless sensing product design.

The ADXL1005 is available in a 5mm × 5mm × 1.8mm LFCSP package, and operates over the -40°C to +125°C temperature range.

- Single, in-plane axis accelerometer with analog output
- Full-scale range: ±100g
- Linear frequency response range: DC to 23kHz typical (3dB point)
- Ultralow noise density: 75µg/√Hz
- Complete electromechanical self test
- Sensitivity stability over temperature within ±5%
- Cross-axis sensitivity: ±1.5% (z-axis acceleration effect on x-axis, y-axis acceleration effect on x-axis)
- Low power consumption: 1mA typical
- Power-saving standby operation mode with fast recovery
- -40°C to +125°C operating temperature range
- 32-lead, 5mm × 5mm × 1.8mm LFCSP (lead-frame chip-scale packaging) package

Analog Devices
www.analog.com

Designing Directions

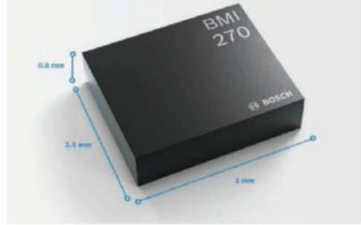
The Asahi Kasei Microdevices AKO9918C is 3-axis electronic compass IC with high sensitive Hall sensor technology realized in a package measuring 0.76mm × 0.76mm × 0.5mm. This makes it 63 percent the size of the conventional AK09916C from Asahi Kasei Microdevices, while maintaining the high performance of the larger-jacketed product.

The AKO9918C's small size and low power consumption make it suitable for inclusion in wearable projects such as smart glasses, watches, VR/AR headsets, and other devices that are sensitive to component footprint, weight, and power usage. Applications for the AKO9918C's features include navigation while the user is walking, running, or hiking; keeping AR applications in proper orientation with physical surroundings; and enabling navigation for autonomous mobile robots.

The package is a pin pitch 0.4mm, 4-pin WL - CSP (Wafer-level chip scale packaging) with power supply on two pins and two pins for I²C communication.

- 3-axis magnetometer device suitable for compass application
- Built-in analog to digital converter for magnetometer data out
- 16-bit data out for each 3-axis magnetic component
- Magnetic sensor overflow monitor function
- Internal clock source oscillator
- Power on Reset circuit
- Self-test function with internal magnetic source
- Built-in magnetic sensitivity adjustment circuit

AsahiKASEI
www.akm.com



Counting Inertia

The Bosch Sensortec BMI270 is an ultra-low power smart IMU (inertial measurement unit) optimized for space-restricted wearable and hearable applications.

The IMU combines acceleration and angular rate measurement with intelligent on-chip motion-triggered interrupts. The BMI270 includes intuitive gesture, context, and activity recognition with an integrated plug-and-play step counter.

This Bosch unit features 6-axis measurement that can be used to trigger on either motion of various types (any motion, strenuous motion, intermittent motion or continuous) as well as lack of motion. The applications include fitness and movement trackers, smart clothing, and AR/VR devices.

- 2.5mm × 3.0mm × 0.8mm³ 14 pin land-grid array (LGA) package
- Output data rates: 25Hz—6.4kHz (gyroscope) and 0.78Hz—1.6kHz (accelerometer)
- Programmable low-pass filter
- Built-in power-management unit
- Rapid start-up time: 2ms
- Freely configurable secondary digital interface
- 2KB on-chip FIFO buffer for acceleromter, gyroscope, timestamps, and AUX sensor data
- Fast-offset error compensation for accelerometer and gyroscope
- HW synchronization of accelerometer, gyroscope, and AUX sensor
- Two independent programmable I/O pins for interrupt and synch events

Bosch
www.bosch-sensortec.com

DATASHEET URLS:

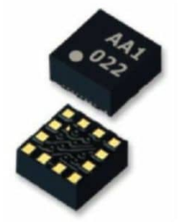
Analog Devices: https://www.analog.com/en/products/adxl1005.html

AsahiKASEI: https://www.akm.com/us/en/products/electronic-compass/lineup-electronic-compass/ak09918c

Bosch Sensortec: https://www.bosch-sensortec.com/media/boschsensortec/downloads/datasheets/bst-bmi270-ds000.pdf

DATASHEET

Datasheet: MEMs Sensors



Measuring Acceleration

The Rohm Kionix KXTJ3-1057 is a tri-axis ±2g, ±4g, ±8g, or ±16g silicon micromachined accelerometer. It is delivered in a 2 × 2 × 0.9mm LGA plastic package operating from a 1.7V – 3.6V DC supply. This small MEMS device is user-configurable to respond to ranges of ±2g, ±4g, ±8g, or ±16g. With its variable sensitivity and multi-axis measurements, the KXTJ3-1057 is suitable for applications that require precise and rapid-response motion measurement without the need for the same very small packaging standard for wearable designs.

Typical of a MEMS device, the sensing element is hermetically sealed at the wafer level by bonding a second silicon lid wafer to the device using a glass frit. A separate ASIC packaged with the sensing element provides signal conditioning and digital communications. The I²C protocol is used for communication to configure the part and monitor outputs. Those outputs are highly configurable to

- Small footprint: 2mm × 2mm × 0.9mm LGA 12-pin package (pin-to-pin compatible with KXTJ2)
- Low current consumption: 0.9µA in standby, 10µA in Low Power, and 155µA in high resolution modes
- Extended user-configurable g-ranges: ±2g, ±4g, ±8g, ±16g
- 8-bit, 12-bit, and 14-bit resolution modes
- Wide supply voltage range: 1.71V to 3.6V with internal voltage regulator

Rohm
rohm.com

Under Pressure

The NXP MPXV7002 series piezoresistive transducers are monolithic silicon pressure sensors. That means it provides an accurate, high-level analog output signal proportional to the pressure applied to the sensor. The sensor has a maximum pressure of 75kPa (kilopascals) which, for those that don't grok metric, is equivalent to about 18.88psi.

The MPXV7002 is an on-chip signal-conditioned, temperature-compensated, and calibrated device for accuracy across a wide operating temperature range. It can work with either fluids or gasses as the measured medium, but it is calibrated using air, so compensation (which is readily accomplished) must be applied for liquid applications.

According to NXP, the MPXV7002 is intended for use in designs featuring separate microcontrollers or microprocessors for data analysis and processing. Packaging for the family of MEMS devices includes a variety of input and orientation configurations. Typical applications include health and medical such as hospital beds and respiratory systems, as well as HVAC and process control deployments.

- Thermoplastic (PPS) surface mount package
- Temperature compensated over +10°C to +60°C
- Patented silicon shear stress strain gauge
- Available in differential and gauge configurations

NXP Semiconductors
www.nxp.com

Keeping the Time

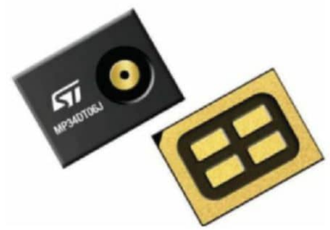
The SiT5008 TCXO is a silicon MEMS-based temperature-compensated crystal oscillator (TXCO) that brings reliability along with vibration and shock environmental resilience. It is a 26MHz oscillator that is able to provide any frequency between 10MHz and 60MHz, accurate to six decimal places.

The SiT5008 is pin-compatible with traditional quartz TCXOs, making it a drop-in replacement on circuit boards for engineers who want to shift designs from traditional quartz to temperature-compensated oscillators. Temperature compensation is important in applications requiring high-precision timing: traditional quartz oscillators can see their vibration rate change with shifts in temperature, leading to inaccurate timing as ambient or operating temperatures change.

Suited for communications, GPS-based applications, and security, the SiT5008 has features that make it an easy option for designers and engineers who want to boost timing accuracy and stability.

- ±2ppm frequency stability (0°C to 80°C)
- Operating temperature from -40°C to 85°C
- Low power consumption of 3.5mA typical at 1.8V
- Standby mode for longer battery life
- Fast startup time of 5ms
- LVCMOS/HCMOS compatible output
- Two Industry-standard packages: 2.0mm × 1.6mm, 2.5mm × 2.0mm

SiTime
www.sitime.com



Tiny Mic

The STMicroelectronics MP34DT06J is a compact, low-power, omnidirectional, digital MEMS microphone. The sensor combines a sensing element capable of detecting acoustic waves with an IC interface manufactured using a CMOS process that allows a dedicated circuit to provide a digital signal externally in PDM format.

The MP34DT06J is a low-distortion digital microphone with a 64dB signal-to-noise ratio and –26dBFS ±1dB sensitivity.

It is available in a top-port, SMD-compliant, EMI-shielded package and is guaranteed to operate over an extended temperature range from -40°C to +85°C.

The MP34DT06J is suitable for use in a wide variety of applications including mobile terminals, notebook computers, portable media players, gaming and virtual reality input devices, and digital cameras.

- Single supply voltage
- Low power consumption
- AOP = 122.5dBSPL
- 64dB signal-to-noise ratio
- Omnidirectional sensitivity
- –26dBFS ±1dB sensitivity
- PDM output
- HCLGA package
- Top-port design
- SMD-compliant
- EMI-shielded
- ECOPACK, RoHS, and "Green" compliant

STMicroelectronics
www.st.com

In its Depths

The TE Connectivity MS5837-30BA is a high-resolution pressure sensor with a water depth resolution of 2mm for depth measurement systems that features an I²C bus interface for external control and communications. The MEMS unit provides a precise digital 24-bit pressure and temperature value delivered through different operation modes that allow the user to optimize for conversion speed and current consumption.

A high-resolution temperature output allows the implementation of in-depth measurement systems and thermometer function without any additional sensors.

- Ceramic-metal package, 3.3mm × 3.3mm × 2.75mm
- High-resolution module 0.2mbar
- Fast conversion down to 0.5ms
- Low power, 0.6µA (standby < 0.1µA at 25°C)
- Integrated digital pressure sensor (24-bit ΔΣ ADC)
- Supply voltage 1.5 to 3.6V
- Operating range: 0 to 30 bar, -20 °C to +85 °C
- I²C interface
- No external components required
- Excellent long-term stability
- Hermetically sealable for outdoor devices
- Gel-filled protection & stainless steel cap for corrosion resistance
- Provides both pressure and temperature readings

TE Connectivity
www.te.com

Tracking Your Moves

The TDK InvenSense ICM-42688-P is a 6-axis MEMS motion tracking device that combines a 3-axis gyroscope and a 3-axis accelerometer. The device includes 20-bit data format support in FIFO for high data resolution. This FIFO format encapsulates 19 bits of gyroscope data and 18 bits of accelerometer data.

Features include an InvenSense on-chip APEX Motion Processing engine for gesture recognition, activity classification, pedometer, programmable digital filters, and an embedded temperature sensor. To get information out of the ICM-42688-P, there is a configurable host interface that supports I³C, I²C, and SPI serial communications.

- User-programmable digital filters for gyro, accel, and temp sensor
- Noise and current consumption
- APEX Motion Functions: Pedometer, Tilt Detection, Tap Detection
- Wake on Motion, Raise to Wake/Sleep, Significant Motion Detection
- Host interface: 12.5MHz I³C, 1MHz I²C, 24MHz SPI
- Gyroscope Noise: 2.8mdps/√Hz & Accelerometer Noise: 70µg/√Hz
- Low-Noise mode 6-axis current consumption of 0.88mA
- User selectable Gyro Full-scale range (dps): ±15.6/31.2/62.5/125/250/500/1000/2000
- User selectable Accelerometer Full-scale range (g): ±2/4/8/16

TDK InvenSense
invensense.tdk.com

DATASHEET

DATASHEET URLS:

Rohm: https://fscdn.rohm.com/kionix/en/datasheet/kxtj3-1057-e.pdf

NXP Semiconductors: https://www.nxp.com/docs/en/data-sheet/MPXV7002.pdf

SiTime: https://www.sitime.com/support/resource-library/datasheets/sit5008-datasheet

DATASHEET URLS:

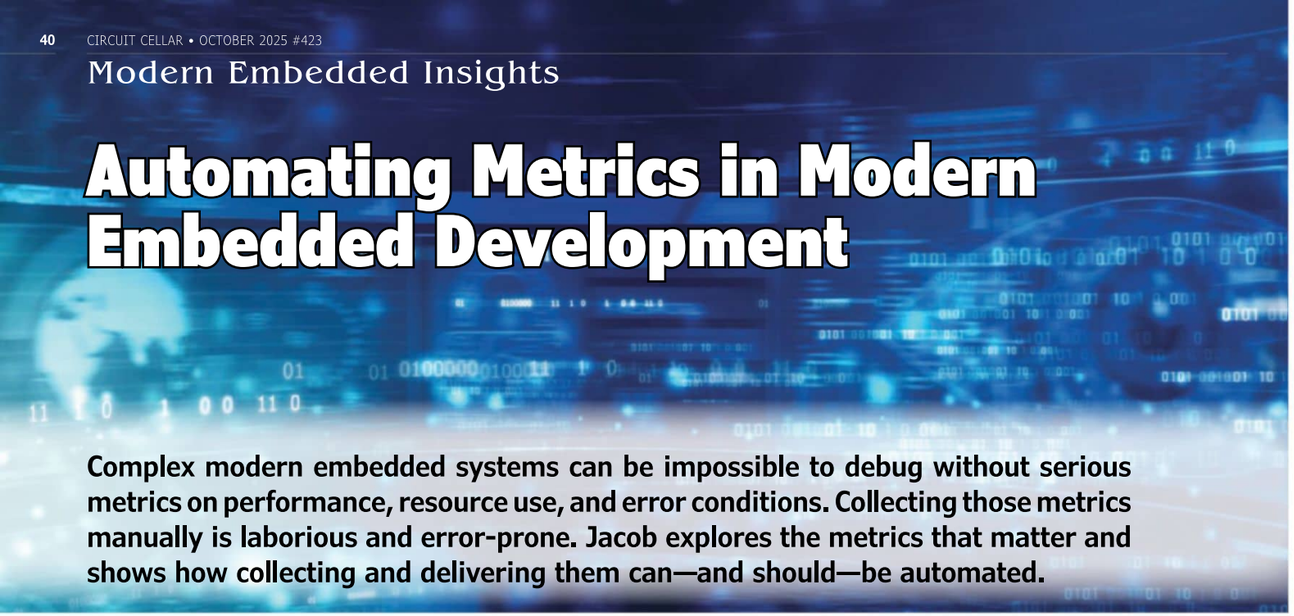
STMicroelectronics: https://www.st.com/resource/en/datasheet/mp34dt06j.pdf

TE Connectivity: https://www.te.com/en/product-MS583730BA01-50.html

TDK InvenSense: https://invensense.tdk.com/download-pdf/icm-42688-p-datasheet

Modern Embedded Insights

Automating Metrics in Modern Embedded Development



Complex modern embedded systems can be impossible to debug without serious metrics on performance, resource use, and error conditions. Collecting those metrics manually is laborious and error-prone. Jacob explores the metrics that matter and shows how collecting and delivering them can—and should—be automated.

By Jacob Beningo

Modern embedded systems are notoriously complex: hundreds of small modules, countless configuration options, and hardware constraints that affect every decision. Even worse, companies demand more features within tight deadlines, making handling a complex system a stressful task.

As we keep adding features, we lose confidence in whether our system is stable or functionally correct. The sheer complexity makes us doubt the quality of the work we deliver. We often ask ourselves: Will our changes crash the system in the next deployment? We essentially work under a shadow of fear.

And yet many embedded teams still work without real visibility into their code's health. They rely on intuition, firefighting, and gut feel to decide when to refactor or where quality might be slipping.

The problem is, without data, these decisions are often wrong, and the cost shows up as defects, delays, and brittle architectures.

That's why the best embedded teams don't just write software. They measure it. Metrics turn vague hunches into concrete trends, letting you spot risk before it snowballs and steer your development toward maintainable, high-quality code.

In this post, we'll explore why metrics matter, the key ones every team should track, and how to automate them directly into your build system—so measuring quality becomes as routine as compiling code.

WHY METRICS MATTER IN EMBEDDED DEVELOPMENT

In 2007, two elderly ladies in Oklahoma were driving somewhere in their Toyota Camry. Their trip ended tragically since one of them was killed and the other seriously injured in a crash.

The investigation revealed that a defect in the vehicle's electronic throttle control system caused it to accelerate uncontrollably. The car's code also revealed a grim reality: the firmware was extremely complex—often described as "spaghetti code"—and contained:

- 11,000 global variables
- 80,000 coding standard violations
- 67 untestable functions

This was Toyota's infamous unintended acceleration case. And that's just one example when we don't manage our code complexity and architecture.

It's hard to just rely on intuition or gut feeling to tame the project's complexity. That's why we need an objective and easy way to see the insights from our code: metrics.

Without metrics, you're flying blind. There's no way to track quality trends, no feedback loop on architecture, and no early warnings. Metrics can help answer questions such as:

- Is our codebase getting harder to maintain?
- Are we adding risk with each release?
- Where should we focus the refactoring effort?

Visualizing these trends gives leadership—and especially developers—the insight to act before defects accumulate. Had Toyota used metrics for its car's firmware code, it could probably have anticipated (or even prevented) such an accident.

However, metrics alone wouldn't have solved the problem. To make metrics more effective, several guidelines need to be followed:

- **Start with what can be measured now**. Add more later as tools and processes mature.
- **Focus on trends, not the absolute metric value**. A single result is less valuable than a trend.
- **Review results regularly** as part of team meetings.
- **Use results to guide action**. Metrics should support decisions, not assign blame.

KEY METRICS FOR EMBEDDED TEAMS

Not all metrics are equally valuable. Select those that give actionable information about maintainability, reliability, and risk.

Here are some useful metrics to track.

Cyclomatic Complexity: Cyclomatic complexity measures the number of independent paths through a function. A higher score means more potential execution paths, which increases the effort required for testing and maintenance.

Guidelines:

- Complexity score 1-10: Low. Generally easy to understand and test.
- Complexity score 11-20: Moderate. Review before adding more features.
- Complexity score above 20: High. Split or simplify the function.

Track complexity per function. Track trends per module. Review the top offenders in each release. Tie refactoring work to the worst scores first.

Tool to use: *pmccabe*. More on this later.

Static Analysis Warnings: Static analysis tools detect potential issues without running the code. Common results include:

- Unused variables
- Unreachable code
- Possible null pointer dereferences
- Buffer overflows
- Mismatched memory allocation and deallocation
- Resource leaks

Resolve warnings quickly. You probably want to suppress some warnings because resolving them doesn't make sense. In that case, suppress only after review. Keep a suppression file under version control with clear comments.

Tool to use: *cppcheck*.

Code Coverage: Code coverage measures the percentage of code executed by automated tests. Coverage includes line, function, and branch. Each aspect reveals different risk areas.

- Line coverage shows which lines execute.
- Function coverage shows which functions execute.
- Branch coverage shows which decision outcomes execute.
- Track coverage at the module and system levels. Prefer stable or rising trends. Review uncovered files when adding features or fixing defects.

Tools to use: *gcov* or *gcovr*.

Be warned, though! 100% code coverage doesn't mean your system doesn't have bugs! It just means that your test cases execute every line of code. There can be poor test cases, logical errors, and many other problems.

Code coverage, while good to track, is often misleading and more a vanity metric. There are also other metrics worth exploring, such as:

- MISRA/CERT compliance
- Architecture Metrics like LCOM (Lack of Cohesion), CBO (Coupling), RFC (Class Responsibility)

While those are valuable, in this article, we'll focus on two foundational ones—***Cyclomatic Complexity*** and ***Static Analysis***—because they're quick to set up, easy to understand, and immediately actionable.

In the next section, we'll:

- Install and use pmccabe (complexity analysis) and cppcheck (static analyzer) manually.
- Integrate them into CMake as dedicated build targets.

```
beningo@Beningos-MacBook-Pro Application % pmccabe -v packet.c
Modified McCabe Cyclomatic Complexity
|   Traditional McCabe Cyclomatic Complexity
|   |     # Statements in function
|   |     |   First line of function
|   |     |   |   # lines in function
|   |     |   |   |   filename(definition line number):function
|   |     |   |   |   |
7   7     18  233 67      packet.c(233): Packet_DecodeSm
2   2     4   335 10      packet.c(335): Packet_Sync
1   1     2   381 5       packet.c(381): Packet_Version
1   1     1   422 4       packet.c(422): Packet_SourceAddress
1   1     1   462 4       packet.c(462): Packet_DestinationAddress
1   1     1   502 4       packet.c(502): Packet_MessageID
3   3     7   542 24      packet.c(542): Packet_ProcessTwoByteField
5   5     13  602 59      packet.c(602): Packet_Data
1   1     2   697 5       packet.c(697): Packet_Checksum1
1   1     2   738 6       packet.c(738): Packet_Checksum2
2   2     6   780 20      packet.c(780): Packet_EndSync
1   1     1   834 4       packet.c(834): Packet_Get
3   3     36  878 57      packet.c(878): Packet_Validate
1   1     1   962 4       packet.c(962): Packet_ErrorGet
1   1     1   992 4       packet.c(992): Packet_ErrorClear
1   1     13  998 22      packet.c(998): Packet_Encode
1   1     3   1021 6      packet.c(1021): Packet_ResetSm
```

FIGURE 1
A code analysis, including column headers and complexity scores.

COLUMNS

MEASURING CODE COMPLEXITY WITH PMCCABE

pmccabe calculates cyclomatic complexity for each function. It's small, fast, and scriptable.

Installing pmccabe

Install on Debian/Ubuntu:

```
sudo apt update -y
sudo apt install -y pmccabe
```

Install on MacOS via brew:

```
brew install pmccabe
```

Verify the installation by running:

```
% pmccabe -V
pmccabe 2.8
```

Running the Analysis

Run on a source tree:

```
pmccabe *
```

Now as a quick example, let's say we want to analyze the code complexity of a file called `packet.c`:

```
pmccabe -v packet.c
```

The `-v` argument means verbose output, and we use that so the tool will print the column headers. The result would look something like **Figure 1**. As we can see, all functions in the C source code have a complexity score below 10.

As a reminder:

- Score below 10: Good.
- Score 10+: Complex.
- Score 20+: Refactor now.

Tracking these scores over time helps you spot trouble before it grows.

A single "20+" score isn't necessarily a disaster, but a steady climb in average complexity across modules is a warning sign. I can't tell you how many times I analyze a customer's code and find horrific complexity levels simply because they didn't bother to track this simple metric. The largest I've ever seen was 1050+! We often say that complexities of 50 or more are untestable functions!

Once you've identified your top offenders, you can plan targeted refactoring or split oversized functions into smaller, more maintainable pieces.

Still, cyclomatic complexity only tells you *how many* code paths are in your function. It won't catch hidden defects, resource leaks, or buffer overflows. That's why the next step is to pair it with static analysis tools like cppcheck.

STATIC ANALYSIS USING CPPCHECK

cppcheck detects common defects in C and C++ code. It's versatile and easy to automate.

Installing cppcheck

Install on Debian/Ubuntu:

```
sudo apt update -y
sudo apt install -y cppcheck
```

Running cppcheck

Run cppcheck on a source tree:

```
cppcheck --enable=all --inconclusive
  --std=c++17 src/
```

Here's what each argument does:

- `--enable=all` — Enables all available checks, including style, performance, and portability.
- `--inconclusive` — Includes checks that may produce false positives but can catch subtle issues
- `--std=c++17` — Parses the code as C++17. Adjust this to match your project's language standard.
- `src/` — The path to the source code directory you want to analyze.

You can also redirect the output to a file for review:

```
cppcheck --enable=all
  --inconclusive --std=c++17 src/
  2> cppcheck-report.txt
```

Static analysis like this quickly surfaces issues you might miss in code review, such as unused variables, possible null pointer dereferences, buffer overflows, and more.

Now that we've seen how to run pmccabe and cppcheck manually, the next step is to integrate both tools into your CMake build system, so they become part of your standard development and CI/CD workflow.

INTEGRATING METRICS INTO THE BUILD PROCESS

Metrics are most effective when collected automatically, because manual steps fail under time pressure. You should integrate metrics into the build so they run on developer machines and in CI.

Additional materials from the author are available at:
www.circuitcellar.com/article-materials
Reference [1] as marked in the article can be found there.

RESOURCES

TIOBE | www.tiobe.com

RIGOL
Performance That's Easy To See
DM858 Series Digital Multimeter
• 5½ digit reading resolution
• Up to 125 rdgs/s measurement speed
• Up to 500,000 points logging memory
• Measures 11 input signals
• 7" color touchscreen display
DM858
DM858E

Cambrionix
CHARGE • CONNECT • MANAGE
Intelligent USB Hubs For Mass Mobile Device Management
ThunderSync3-16
Supersync15
PowerPad15S
Increase your productivity and ensure reliability;
charge, connect and manage all your mobile devices
using our smart USB hubs and software

pico
Technology
PicoScope® 2000 Series
Like a benchtop oscilloscope, only smaller and better
• 2 channel, 4 channel and MSO models
• 6 instruments in one
• Ultra-compact design
• Up to 100 MHz bandwidth
• Up to 128 MS buffer memory
• Decode over 30 serial protocols as standard
• USB connected and powered
• Windows, Linux and Mac software
From $129

QuantAsylum
Affordable and Powerful Audio Analysis
QA403 24-Bit Audio Analyzer
$590
Free Shipping
• 24-bit ADC/DAC
• Up to 192 kSa/s
• Fully isolated from PC
• 8 input gain ranges (0 to +42 dBV); 4 output gain ranges (-12 to +18 dBV)
• Max AC input = +32 dBV = 40 Vrms. Max DC input = ±40 V, Max ACPEAK + DC = ±56 V.
• USB-powered
• Built-in Attenuator

Not your average web-store...
Your source for technical advice, recommendations and novel products.
Keep checking for our newest listings!
• Saelig Company, Inc. • www.saelig.com • sales@saelig.com • 1-888-7SAELIG • (1-888-772-3544)

Example CMake Integration

Create a dedicated target for analysis:

```
add_custom_target(analyze
    COMMAND cppcheck --enable=all --inconclusive --std=c++17 --xml --xml-version=2 src/ 2>
   reports/cppcheck.xml
    COMMAND pmccabe src/ > reports/pmccabe.txt
    WORKING_DIRECTORY ${CMAKE_SOURCE_DIR}
    COMMENT "Running static analysis and complexity metrics"
)
Add a directory for reports:
file(MAKE_DIRECTORY ${CMAKE_SOURCE_DIR}/reports)
You can then run:
cmake -B build -S .
cmake --build build --target analyze
```

By adding pmccabe and cppcheck into CMake, you make metrics collection a natural part of every build (whether it's on a developer's laptop or a CI pipeline).

INCORPORATING METRICS INTO DAILY WORKFLOWS

Metrics are only useful if they're part of the team's day-to-day development process, not buried in a one-off report. Make them effortless to run. The exact command will depend on your workflow. For example, some teams use a wrapper script like:

```
./project.sh analysis build
```

while others simply run the CMake target directly:

```
make analyze
```

A link to project templates of project.sh and the build system can be found on *Circuit Cellar*'s Article Materials and Resources webpage. [1]

The key is that running metrics should be as easy as building the project. In CI, display the results where everyone can see them. A metrics scoreboard on your pipeline dashboard makes quality trends impossible to ignore.

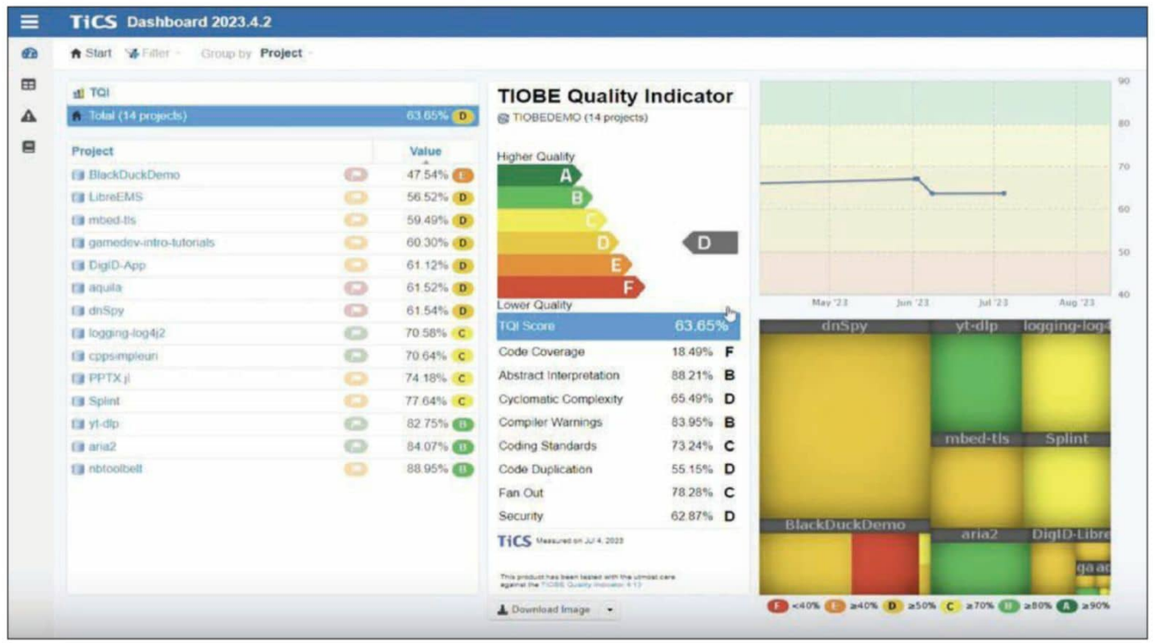


FIGURE 2
Feedback on the TIOBE TiCS' dashboard aggregates various metrics into a continuously updated, interactive view.

For example, the TIOBE TiCS' dashboard aggregates metrics such as complexity and code coverage into an interactive view that updates with every build. **Figure 2**. Tools like this give the whole team immediate, visual feedback on code quality, making it easy to spot trends before they become problems.

Finally, focus on trends, not single values. When metrics are automated, visible, and reviewed regularly, they stop being an afterthought and start guiding engineering decisions.

DATA THAT WORKS

In modern firmware development, if you're not measuring quality, you're managing risk by gut feel. That's how complexity quietly piles up, how subtle defects slip past review, and how teams lose confidence in every release.

We've seen where that road can lead. In Toyota's unintended acceleration case, spaghetti code and a lack of objective quality checks contributed to a tragic outcome (among other things). While most projects won't carry that level of human risk, the pattern is the same: without visibility, problems grow in the dark.

By integrating metrics such as cyclomatic complexity and static analysis into your build system, you gain the data to act early and decisively. That unlocks:

- **Faster refactoring decisions**—You know exactly where the worst complexity lives.
- **Earlier bug detection**—Issues surface before they hit production.
- **Long-term maintainability**—Trends guide architecture and prevent decay.

If you've already modernized your build system, don't stop short. Add metrics. Automate insights. Build smarter.

ABOUT THE AUTHOR

Jacob Beningo is an embedded software consultant and founder of the Embedded Software Academy, where he helps engineers master modern practices to design efficient, scalable embedded systems. He helps developers stay ahead in an evolving industry with hands-on training, expert insights, and strategies. He's consulted and trained companies such as Arm, General Motors, NASA, and others, completing more than 200 projects. Jacob has published books on embedded software development techniques. He holds three degrees, including a master's in space systems engineering from the University of Michigan. You can learn more about Jacob's work at www.beningo.com.

life.augmented
ST's high-accuracy L9961 Battery Management System controller makes lithium batteries perform better and last longer by providing monitoring, balancing, and protection for industrial applications as cordless power tools, energy-storage systems, portable equipment, and more.
L9961
www.st.com

Start to Finish

Audio Power Amplifiers

Choosing an Audio Amplifier for Your Design

By
Stuart Ball

We don't think of audio amplifiers as part of embedded systems, but if you need an audio output, whether for an MP3 player, a warning annunciator, a technician-to-patient voice system in a medical device, or a verbal output for a GPS device, selecting the correct audio components is important.

An audio output consists of several components, starting with something that generates an audio signal, waveshaping and amplifying the signal, and then drives the output device, whether it's a speaker, headphone jack, or other transducer. In this article I'm only going to look at the final output stage: the audio power amplifier.

The output transducer for an audio system is typically a speaker or similar low impedance device. Audio transducers are typically speakers and headphones (which are tiny speakers) with an impedance ranging typically from 2Ω to 32Ω but going up to 600Ω. Due to this, producing usable (that is, audible) output requires an amplifier capable of driving more current into the transducer than is needed for normal signals. For example, to produce 1W of power into an 8Ω speaker requires applying Ohm's power law like this:

$P = I^2R = V^2/R.$
$1W = I^2 \times 8\Omega, I = 350mA.$
Similarly, $V = 2.8V$

Specifically, producing 1W into an 8Ω speaker requires 2.8V at 350mA. I calculated these as DC values, but the same principle applies for an AC audio signal. Although generating 2.8V from a 5V or 3.3V embedded system isn't too hard, the 350mA requirement demands more current capability than an MCU output can provide. The audio power amplifier translates a low-level audio signal into sufficient power to drive the speaker (or other transducer).

AMPLIFIER BASICS

Power amplifiers are categorized into five classes: A, B, AB, C, and D. The differences are in how the output power is generated, how the output drive devices are biased, and the efficiency.

Class A amplifiers, although common in the small signal side, are rarely used in audio power applications other than some audiophile equipment; the efficiency is too low. Class C amplifiers are primarily used in RF power stages. Therefore, I won't be covering those two types here.

A Class B amplifier is illustrated in **Figure 1**. The two transistors essentially act as emitter followers, and the output voltage follows the input voltage. Either the NPN or PNP transistor is conducting; the other is in cutoff. This is about the simplest form of the Class B, and it's included mainly to illustrate the concept of crossover distortion. If you look at the input and output waveforms, you will see a discontinuity in the output waveform as the signal goes through 0V. This is because the output transistors won't conduct until the base-emitter voltage is about 0.7V (for silicon transistors), so no amplification happens.

The Class B amplifier has the advantage of not dissipating power when there is no signal but does have audible distortion at low signal levels. Note that the figure uses both positive and negative supplies; in many MCU applications, there will be a single positive supply, the negative supply is ground, and there is a large capacitor in series with the speaker. But the principle of operation is the same.

Figure 2 shows the Class AB amplifier, again about the simplest form you can draw. The Class AB is a Class B with a pair of diodes to compensate for the base-emitter voltage, and two resistors to provide a little bit of current through the transistors. The diodes match the base-emitter voltage of the transistors (silicon diodes for silicon transistors), and the amplifier will always use some current even when no signal is applied. There are other ways to build a Class AB amplifier, such as more sophisticated biasing techniques, but the general principle is the same; to avoid crossover distortion, some current flows through the output transistors even when no signal is applied.

In a high power amplifier, or in an IC that contains a Class AB output, the diodes will be thermally connected to the transistors so that their forward voltage tracks that of the transistors. The schematic shows PNP and NPN driver transistors; the amplifier can also be designed with two NPN output transistors.

When operating, the Class B and Class AB amplifiers dissipate power in the drive transistors. For example, say you have a 12V power supply and the amplifier is driving 6V into an 8Ω speaker. The speaker is dissipating 4.5W when the voltage is at 6V. Since the transistor (we'll say it's the NPN in this example) is in series with the speaker, the transistor current is the same as the speaker current, and the voltage across the transistor is half the supply voltage, so the transistor power dissipation is also 4.5W during the instant that the speaker voltage is 6V. So, in this example, the power supply has to produce 9W (the speaker dissipation plus the transistor dissipation) and only half of that is used to drive the speaker; the other half is wasted as heat in the transistor.

I've been using DC values for these examples; in a real audio system, the signal is a mix of sine waves. For a pure sine wave, the average power dissipated in the transistor or speaker is the root mean square (RMS) of the peak signal. Therefore, if that 6V speaker voltage is a 1000Hz sine wave, with a peak voltage of 6V, the transistor and speaker will dissipate 4.5W each at the peak. But the RMS voltage is 6V/ x 0.707, or 4.24V. That's the average voltage applied across a purely resistive load by a pure sine wave signal.

If the signal is 3V peak, then the speaker is dissipating 1.125W during the instant that the voltage is at the peak. But the voltage across the transistor is 12V – 3V, or 9V. The current is the same as the speaker, or 375mA, so the transistor is dissipating 3.4W more than the speaker. Of course, the amplitude of a real audio signal varies over time, so the instantaneous RMS voltage and power also vary. Since only one transistor at a time is conducting, each transistor will be dissipating only half of the total "wasted" power.

Figure 3 shows a simple Class AB amplifier, using the Texas Instruments (TI) LM4871 amplifier. The LM4871 can produce up to 1.5W into an 8Ω load [1]. The amplifier uses a bridged output, driving both sides of the load with out-of-phase signals, as opposed to a single-ended output that drives one side of a

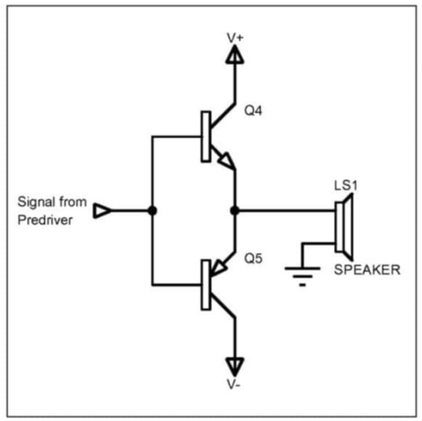


FIGURE 1
Simple Class B amplifier schematic.

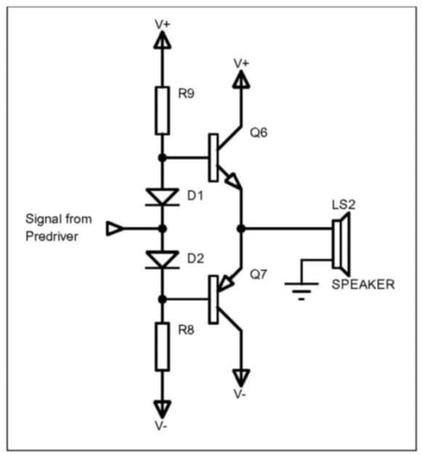


FIGURE 2
Simple Class AB amplifier schematic.

load with the other side grounded. Thus, the instantaneous voltage across the load is twice the voltage you would measure from ground to one output.

For this demonstration circuit, I used a 10Ω resistor as a load. Adjusting the signal generator for 1V P-P across the load, the peak power is 1²/10, or 100mW. The RMS power is the peak power x 0.707 (1/√2), so the output power is 70.7mW. Input current was measured with a DVM, and the actual voltage was measured after the ammeter to calculate the input power. The post-meter voltage was 4.82V (0.18V lost across the meter) and the current was 55mA, so the average input power was 265mW, making the efficiency 26%. Consequently, 70.7mW was dissipated in the load and 194mW was dissipated as heat in the device.

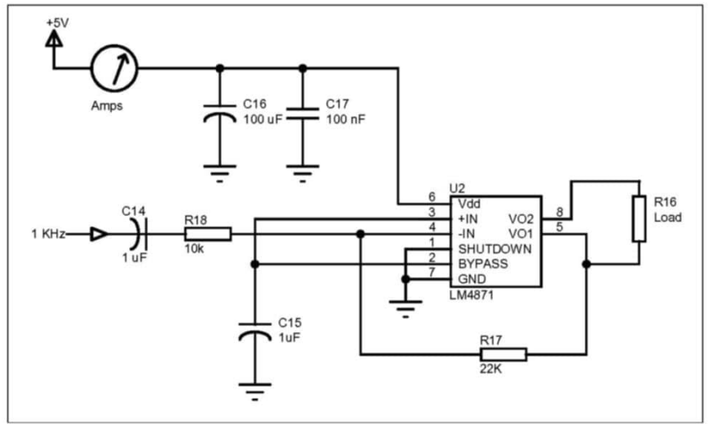


FIGURE 3
Practical Class AB amplifier using LM4871.

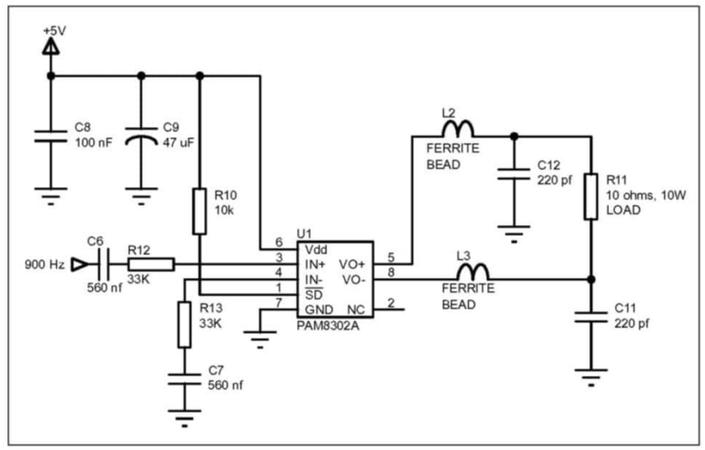


FIGURE 4
PAM8032 Class D amplifier.

Changing to 2.5V P-P across the load resulted in 440mW (RMS) in the load, input current of 141mA, input voltage of 4.58V, for input power of 645mW. This makes the efficiency 68%. But note that even though the efficiency is higher than the last example, the transistors are dissipating more power (205mW). For any instantaneous current through the load, the current in the conducting transistor is the same. The power in the transistor is supply voltage minus load voltage, divided by instantaneous current.

For a Class AB amplifier, you have to figure out what to do with waste heat. A low power IC amplifier such as the LM4871 doesn't have much surface area to dissipate heat. I used a SOIC package for these tests, but the LM4871 is also available in a WSON package that has a copper heatsink pad that can be soldered to a copper area on the PCB to make a heatsink, allowing greater power output.

The point is that using a Class AB amplifier, especially one in a small IC package, requires some level of thermal analysis that accounts for the environment where the device will be operating. Refer to my article "Why's It So Hot in Here?" in the August, 2023, *Circuit Cellar* issue for more information about thermal analysis and heatsinking. [2]

FAST-SWITCHING AMPS

Figure 4 shows a simplified version of a Class D amplifier based on the PAM8032 device. The PAM8032 amplifier uses pulse width modulation (PWM) or a similar technique to drive the speaker. [3] The goal of a Class D amplifier is the same as with any PWM system: to minimize wasted power and the associated heating.

Understanding Class D amplifiers is based on understanding how switch mode devices work. The general idea is that the drive transistors are always on or off. In the on state, they dissipate very little power because the voltage across the switching transistor is close to zero (VI = power). In the full off state, they dissipate no power because the current is zero. Therefore, in a switch mode application, the device being controlled (the speaker in an audio system) is either driven fully on or fully off.

The switching frequency is faster than the device can respond to it, so it responds to the average current. A simple example is an LED; you can control the brightness of the LED by driving it with a constant frequency signal and varying the duty cycle (ratio of the on and off time) to adjust brightness. But the driving frequency must be greater than 20Hz, or a person looking at it can detect the flicker.

Avoiding visible flicker in an LED can be accomplished at a low frequency. But if you connected a 20Hz or 30Hz drive to a speaker you would hear it—at least until the speaker came apart. Since the audio frequency range extends to about 18KHz, using PWM to control audio requires a much higher switching frequency, on the order of 200KHz or more.

The outputs of the Class D amplifier aren't an audio-frequency signal, but are instead square waves that switch between ground and the positive supply rail. It doesn't resemble the input sine signal at all, as shown in **Figure 5**. I captured only one pulse in Figure 5, but the frequency of this specific device is about 233KHz. The duty cycle of the signal determines the audible amplitude from the speaker. The Class D amplifier typically operates at a much higher efficiency than a Class AB amplifier; the PAM8302 can approach 88% efficiency, producing less waste heat.

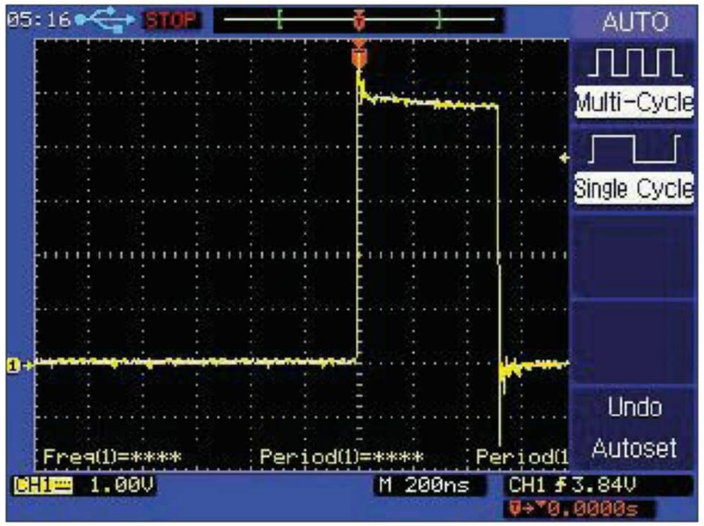


FIGURE 5
Digital output of a Class D amplifier.

EMI

The Class D amplifier does have potential electromagnetic interference (EMI) issues

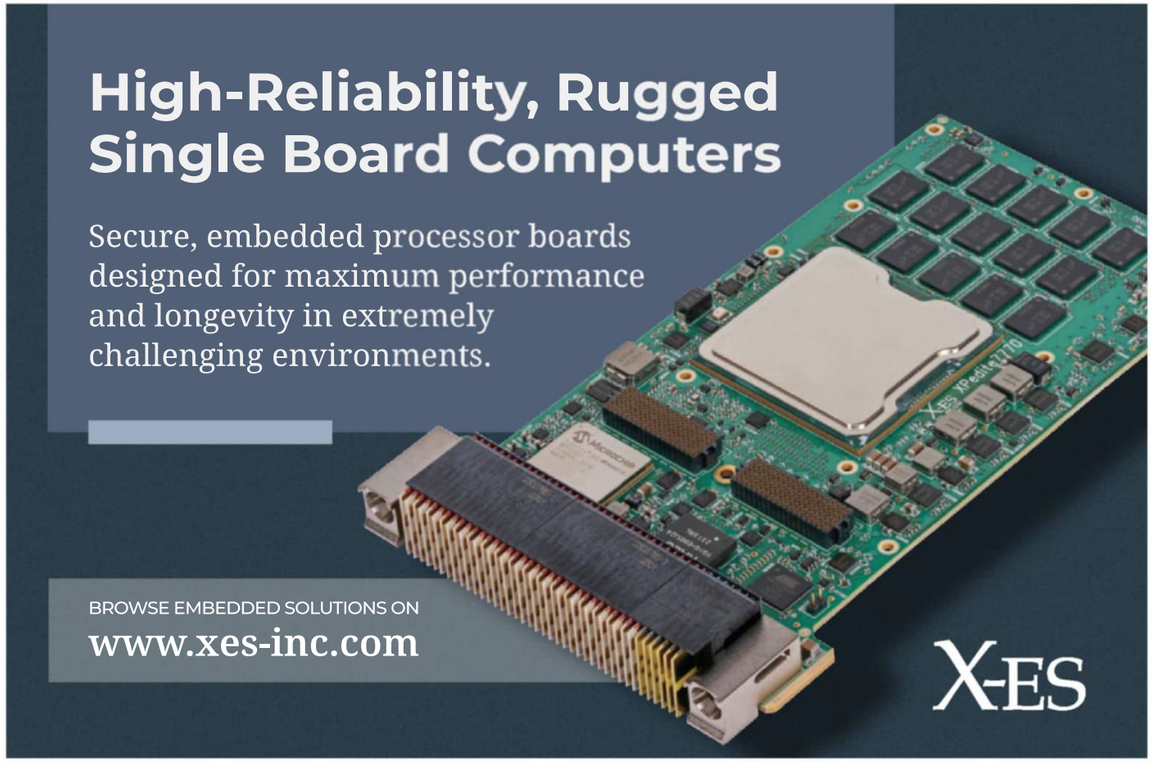
ABOUT THE AUTHOR
Stuart Ball recently retired from a 40+ year career as an electrical engineer and engineering manager. His most recent position was as a Principal Engineer at Seagate Technologies.

Additional materials from the author are available at:
www.circuitcellar.com/article-materials
References [1] to [3] as marked in the article can be found there.

RESOURCES

Diodes, Inc. | www.diodes.com

Texas Instruments | www.ti.com



that the Class AB amplifier does not. Like any PWM device, a Class D amplifier has very fast switching, generating significant harmonics. This is necessary because the output transistors need to be on or off, with minimal time in between the fully on and fully off states. Time between on and off means the transistors are generating heat. The PAM8302 can drive a speaker without any EMI filtering, but filtering may be needed to avoid generating significant electrical noise that can interfere with other devices, such as radio receivers. Figure 4 shows the recommended filtering method for the PAM8302: a couple of ferrite beads and filter capacitors.

The addition of the ferrite beads and capacitors will dampen emissions from long speaker leads. But any PWM system, including a Class D audio amplifier, has high frequency currents in the power supply leads. The average power going to the load may be, say, one watt. But the instantaneous power is the load resistance divided by the power supply voltage. For the circuit in Figure 4, this is $5^2/10$ or 2.5W. The wattage matters here because for a Class D power amplifier, the higher the current (and therefore the higher the wattage), the more emissions is generated. This may be filtered out at the speaker, but it may result in high conducted emissions at the power supply.

These problems can be alleviated by good design: use a PCB with power and ground planes, use adequate DC bypassing, minimize wire length to the speaker, make sure grounds are low impedance, add an input lowpass filter to the amplifier board if needed.

CONCLUSION

Audio amplifiers are not strictly for embedded applications. But some embedded applications do need some form of audio output, whether it's for entertainment, notifications, or warnings. The Class AB amplifier is simple to use and has minimal EMI issues. The Class D amplifier is significantly more efficient and generates less waste heat but may have EMI issues that have to be mitigated. Use whichever one best suits your requirements.

COLUMNS

WHAT IS RMS?

RMS (Root Mean Square) is a way of calculating the equivalent voltage or power of a periodic waveform. "Equivalent" in this case means the DC voltage that produces the same heating in a coil or resistor. The idea is that if you apply an AC voltage with a peak value of X, the RMS value of that voltage, applied as a DC voltage, will cause the same power to be dissipated as the AC signal will. This is important in calculating the effective power delivered to the heating element in an electric oven by the household AC power, or the power delivered to a speaker by an audio signal. Thus, a 15V RMS sine wave driving a resistor will produce the same heating as a 15V DC voltage applied to the resistor.

Calculation: Calculating the RMS value involves taking the square root of the voltages, summing them, calculating the average, and then taking the square root. **Figure 6** shows half of a sine signal that goes from 0V to 1V with samples taken at 0.1 × pi, 0.2 × pi, etc. Values are rounded to two decimal positions. **Table 1** shows the actual values out to four digits of precision. The RMS calculation is performed by summing the squares of the numbers ($0.309^2 + 0.5878^2$....) and dividing by the number of samples (10). The result is 5. Take the square root of that for the RMS value, which is 0.701 x the peak voltage ($\sqrt{2}$) for a sine wave. Note that I did this calculation with just half of the sine wave. The other half would be symmetrical but negative. The number of samples would be doubled, but the negative values would be the same as the positive values (except negative, of course). Squaring the negative values makes them positive. You would end up with 20 squared samples; divide by 20, and the result would be the same.

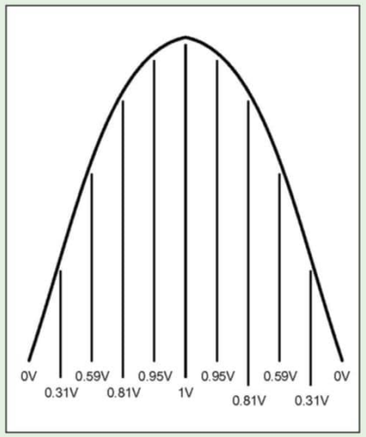


FIGURE 6
half of a sine wave showing voltages at pi/10 intervals

Sine Value
0.0000
0.3090
0.5878
0.8090
0.9511
1.0000
0.9511
0.8090
0.5878
0.3090
0.0000

TABLE 1
Sine values for 1V peak signal

The theoretical RMS value is calculated using integral calculus and really calculates the area under the waveform curve. But for practical applications, the discrete samples are close enough—within the tolerances of the components that make up the circuit, whether it's the resistance of an oven heater or the components that determine the gain of an audio amplifier.

RMS Power: Power is calculated as V^2/R or IR or I^2R. The RMS power is the peak power squared, divided by the impedance, and multiplied by 0.707. Therefore, the RMS power of a 2V peak sine wave into an 8Ω speaker is $(2^2/8)$ x 0.707, or 350mW. Note that this is based on a signal that varies above and below ground. In other words, one side of the speaker (or other load) is driven, and the other side is grounded. In a differential amplifier, such as that used in the article, one side of the speaker is driven high while the other is driven in the opposite direction, so the voltage across the speaker is 2x the single-ended value. A differential amplifier can produce twice the voltage, so 4x the power, from the same power supply voltage as an equivalent single-ended amplifier. This is because either output of a differential amplifier can simultaneously drive the speaker towards the positive or negative rails.

More Complex Waveforms: RMS for speakers and audio amplifiers is calculated using a pure sine wave. Real audio, as from a music source, is a combination of different sine waves at different frequencies and amplitudes. The result is an unknown waveform. You could calculate the RMS value as long as the waveform is periodic, but it gets more complicated and is valid for a snapshot of that audio waveform.

Verilog HDL

With the right tools

such as this book,

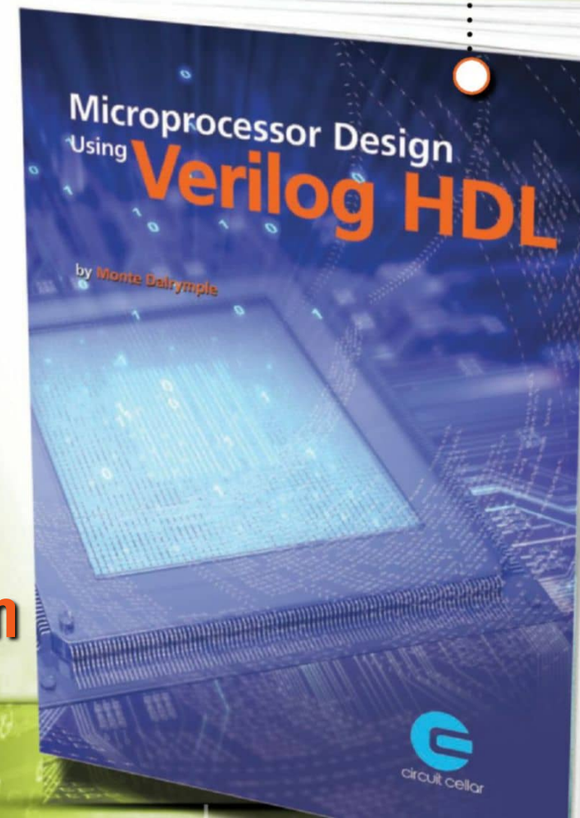
designing a microprocessor can be easy.

Okay, maybe not easy, but certainly less complicated. Monte Dalrymple has taken his years of experience designing embedded architecture and microprocessors and compiled his knowledge into one comprehensive guide to processor design in the real world.

Microprocessor Design Using Verilog HDL will provide you with information about:

- Verilog HDL Review
- Verilog Coding Style
- Design Work
- Microarchitecture
- Writing in Verilog
- Debugging, Verification, and Testing
- Post Simulation and more!

Monte demonstrates how Verilog hardware description language (HDL) enables you to depict, simulate, and synthesize an electronic design so you can reduce your workload and increase productivity.



cc-webshop.com

Embedded in Thin Slices

Binary Fixed-Point Arithmetic

Extracting Signals from Noise and Other Distractions

Speed or precision? That's the basic decision involved in fixed-point versus floating-point arithmetic in computing. Bob looks deeply into the hows and whys of fixed-point arithmetic and explains why both speed and precision are possible with the right technique.

COLUMNS

By
Bob Japenga

When I graduated from college, my first job was with Hamilton Standard (now Collins Aerospace). The part of the company that I worked for made strapdown inertial navigation systems (INS). In those days, most INS consisted of three or four gyroscopes and three or four accelerometers—one for each axis and, optionally, a fourth for error-checking places at 45 degrees to the coordinate system. In conventional systems, the inertial platform was mechanically gimbaled in such a way that the coordinate system remained fixed in space. The inertial platform was kept in a fixed orientation relative to their initial conditions when they were calibrated on the launch pad. The signals from the gyros and accelerometers fed the control system that used the gimbals to keep the inertial platform orientation fixed in space. With no movement of the initial inertial reference frame, the math for navigation was much simpler.

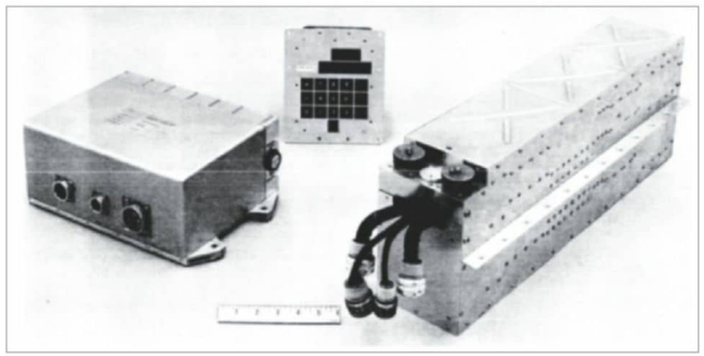


FIGURE 1
Lunar Module Abort Guidance System.

With the advent of computer power, it was possible for the guidance computer to perform the math for navigation with the gyros and accelerometers strapped down to the frame of the spacecraft. The math was intense. Our division's INS were used on the Apollo Lunar Module backup guidance system (called the LM Abort Guidance System, **Figure 1**), the Viking Mars lander, and the Delta launch rocket. On the Apollo program, ours was the only strapdown INS.

I think I got the job because my senior project in aerospace engineering was helping a professor build a strapdown INS. I knew basically nothing about what it was or how it worked, but it was on my resume.

Among others, two memories stand out. We didn't have cubicles but had what we called a "bullpen." There was one large room with 30-50 desks. All programming was done with punch cards. Our simulators were programmed with boxes of punch cards. Unlike in college, there was a separate team in a separate room that put the program on the punch cards. The bigger the box of punch cards on your desk, the higher you were technically in the organization.

The other memory was of large sheets of paper where engineers tracked the binary point (BP) through the complex control algorithms of the strapdown INS. Even though I had strapped down inertial guidance systems on my resume, I had no clue what a binary point was or what BP4 meant on all of those large pieces of paper. A signal went into a comparator with BP4 and came out with BP-2. It was definitely gibberish to me. I hope that by the end of this article you will know what it is and how to use it. This month we look at what fixed-point arithmetic is, why we need it, how it works, and some performance data when compared to floating-point arithmetic.

WHAT IS BINARY FIXED-POINT ARITHMETIC?

Generally, as C programmers, we think of two basic numeric types: integer and float. Thus, any arithmetic that requires a fractional part, we handle as a float (short for floating-point). Thus, if we want to represent π, we would declare it as a float. Fixed-point arithmetic is a way to represent what we typically call a floating-point number as an integer and a way to perform fast fractional arithmetic with integers. You use fixed-point arithmetic when you need to do multiplication and division with numbers that are not integers and do not have the resources to do floating-point arithmetic.

In fixed-point arithmetic, we take a n-bit integer and break it into an o-bit integer and a p-bit fractional part where:

$$n = o + p$$

In binary fixed-point notation, the BP is p or the number of bits assigned to the fractional part. The exact placement of the BP varies based on your desired precision. If you had an 8-bit number and set the binary point at 4 (**Figure 2**), the number 2.5 would be represented by:

$$0010\ 1000_b \text{ or } 2 + 8/16$$

In this article, binary numbers will be indicated by a $_b$ subscript and hexadecimal numbers will be indicated by an $_h$ subscript. To understand this article, you will need to understand binary and hexadecimal notation. An excellent explanation of this can be found on *Circuit Cellar's* Article Materials and Resources webpage. [1]

WHY DO WE NEED IT?

A great question to ask is: Why do I need to know about fixed-point arithmetic? Because the math works faster than software implementations of floating-point arithmetic. Thus, you can use slower processors without floating-point processors (cheaper and less power) to solve the same problems. In addition, even with the great speeds achieved by today's processors, even with floating-point hardware implementations, some applications still use fixed-point arithmetic. It's a tool in our toolbox that every embedded designer needs to know how to use.

What are the downsides? It is more complicated for the designer—remember the large sheets of paper!—and is a more error-prone approach.

PERFORMANCE BENCHMARKS

Benchmarks vary widely, but if you use, for example, the Microchip floating-point library, which can be found on *Circuit Cellar's* Article Materials and Resources webpage [2], a single precision multiply can take more than 20 times longer (140 cycles on a PIC18) than an integer multiply. The addition of a hardware floating-point unit can wipe that performance difference out—and possibly wipe out your recurring cost or power budget.

HOW IT WORKS

Let's walk through a few specific examples of binary fixed-point arithmetic. Decimal fixed-point arithmetic exists, but there is magic in binary fixed-point arithmetic. If we use BP (base 2) arithmetic, multiplies and divides "magically" come out the with correct answer when you adjust the binary point. In addition, when we want to get the answer back in the engineering units of our desired precision (i.e., move the binary point), we need only shift the answer right or left. Later, you will see how this proves useful.

Using an 8-bit number for simplicity, take a number like pi as 3.14 and use 2 bits for the integer portion (thus BP6) as shown in **Figure 3**. You would ***not*** represent it like this:

11_b or 3 (11_b in bits 6 and 7) 001110_b and
14 (001110_b in bits 0-5)

If you did that, you could not do simple math transformations. Rather you would represent it as:

11_b or 3 and 001001_b or
9/64 or $9/2^6$ (where 6 is the BP) or 0.140625

Combine them, and you would have 11001110_b or CD_h in hexadecimal notation. 3.140625—not exactly pi, but close.

When fractional numbers are represented this way, addition, subtraction, multiplication and division work seamlessly. Let me illustrate that.

Let's say you wanted to have three places of precision beyond the decimal point with number 23.478. To have one part in 1000 precision would require a BP10 (2^{10} = 1024).

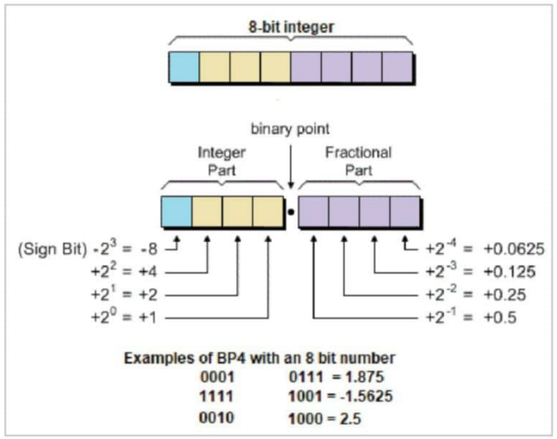


FIGURE 2
BP 8-bit integer.

COLUMNS

This means there would be 10 binary digits to the right of the decimal point. If we had an 18-bit number like we had on Apollo, we would have:

for the integral part:
For 001 0111_b or decimal 23 for bits 16-10 (bit 17 is the sign)
AND
for the fractional part:
For 01 1110 1001_b or decimal $489/2^{10}$ or 0.4775390625

If we put that together into an 18-bit word (where bit 17 is the sign), we get:

00 0101 1101 1110 1001_b

If we want to multiply this by 2, we could shift everything left by 1 and we would get:
00 1011 1011 1101 0010_b
OR
46 + .955 (978/1024), which is close to 46.956 (We lose precision because $489/2^{10}$ = 477.5390625, not 478.

However, our navigation equations rarely multiply by 2. Imagine we are multiplying two 8-bit numbers (for ease of our calculations). And let's say that we need precision to one part in 10 and thus we will use BP4. And we are multiplying:

2.7 times 3.2—both BP4s. The answer should be 8.64.
2.7 would look like:

0010_b or 2 + 1011_b or .6785 ($11/2^4$)
put together in hexadecimal notation as: $2B_h$

And 3.2 would look like:
0011_b or 3 + 0011_b or .1875 ($3/2^4$ = .1875)
or put together as 33_h

The 16-bit answer to $2B_h$ X 33_h would be represented as an integer with BP8 (BP4 + BP4):

0000 1000_b 1001 0001_b

And here is where the magic happens:
8 + .56640625 (141/256 or $141/2^8$)

And if we need to have the answer back in BP4, we just right shift by 4 and get:

1000 1001_b or 8 + .5625 (9/16 or $9/2^4$)

Additional materials from the author are available at: www.circuitcellar.com/article-materials
References [1] to [5] as marked in the article can be found there.

RESOURCES

MathWorks | www.mathworks.com

Texas Instruments | www.TI.com

Obviously, we have lost some precision, but we lost it quickly! One 70 microsecond multiply and 25 microseconds (3*4 + 13) to multiply and normalize on our Apollo computer. The Programmer's Reference and more information on the LM Abort Guidance System can be found on *Circuit Cellar's* Article Materials and Resources webpage. [3]

If we represent the calculation with two 18-bit binary point numbers, we can greatly increase the precision. Let's instead represent both numbers as BP12 in an 18-bit integer.

2.7 would look like:

00 0010_b or 2 + 1011 0011 0011_b + 0.699951171875 ($2867/2^{12}$)
$02B33_h$

3.2 would look like:

00 0011_b or 3 + 0011 0011 0011_b or 3 + 0.199951171875 ($819/2^{12}$) 3333_h

Performing an integer multiply on this would result in 0 $08A3\ C429_h$ as a binary point 24 in a 36-bit word or 8 + ($A3C429_h/2^{24}$) or 0.6397119164466858—which is not too far off from 8.64!

An integer divide works exactly the same way, except that we subtract the two binary points.

Let's do a simpler example with 8-bit numbers for division:

2.5 divided by .5 with both at BP4 should yield 5.0.
2.5 would look like:
0010_b or 2 + 1000_b or $8/2^4$ = .5

And 0.5 would look like:
0000_b or 0 + 1000_b or $8/2^4$ = .5)

The result would be:
0000 0101 using the 73-microsecond division on the Apollo computer.

What do you think the BP would be? If you said zero, you would be correct. With division you subtract the divisors BP from the dividends BP. The answer would be:

5 BP0

In summary, once you have determined the precision you need (more on that later), and divided your number into an integer portion and a fractional portion, put the integer portion in the upper bits and take the fractional portion and multiply it by 2^{BP} and place that number in the lower bits of your integer. Now you can add and subtract your integers with identical BPs. When you multiply, just add the BPs together. When you divide, you subtract.

To normalize any number (for example to add or subtract like BPs), a simple shift left or right will accomplish this.

You can represent big numbers (numbers bigger than you can represent in your integer) with negative BPs. Using our 8-bit number, 0000 0101_b with BP-1 would be 10. With BP-2, it would be 20, and so on. 0111 1111_b with BP-1 would be 256. With BP-10 it would be 131,072. In effect there are no fractional parts—only integers. With negative BPs, everything works the same with additions and subtractions (no change to the BP), multiplies (add the BPs), and divides (subtract the BP).

One of the complications that must be avoided is overflow. In your analysis (those big sheets of paper), you must make sure that you know the upper and lower limits of your inputs. Overflow is such a serious issue in a control equation that the LM Abort Guidance System had an overflow indicator to help the programmer who had not completed his or her analysis correctly.

I hope these examples demonstrate both the power and the complexity of using binary fixed-point arithmetic.

DETERMINING WHERE TO PUT THE BINARY POINT

To maximize precision for a given equation, you want each operation to maximize your word size without overflow and minimize the number of shifts. Thus, since the integer is easy to represent, you can pick the smallest number of bits to represent the integral part of your variable. This is tricky and complicated. Even high-level languages I have used that have fixed-point support, such as Ada and JOVIAL, don't do that work for you. There are some wonderful tools to help minimize quantization errors and maximize precision; however, that is beyond the scope of this thin slice. An excellent transcript of a lecture discussing quantization and precision can be found on *Circuit Cellar's* Article Materials and Resources webpage. [4]

WHAT ARE QUANTIZATION ERRORS?

Quantization errors occur with both fixed-point and floating-point arithmetic. Because our software represents real numbers in a finite number of bits, you encounter the errors we have already demonstrated. If you have ever done a spreadsheet calculation and were expecting $23.00 and got $22.9999997664, you have seen floating-point quantization errors. Fixed-point quantization errors are typically constant across the full range of a number, and floating-point quantization errors increase with the number's magnitude. An excellent analysis of quantization errors can be found on *Circuit Cellar's* Article Materials and Resources webpage. [5]

FIXED-POINT BEAUTY

As embedded designers, knowing that we can perform some pretty complicated math with some pretty low power processors is a valuable tool for your toolkit. I regret that I was snowed by the large pieces of paper and didn't learn the beauty of binary fixed-point arithmetic until after I designed some pretty kludgy control equations. Binary fixed-point arithmetic can provide several advantages in addition to faster calculations and using less power:

- Predictable execution times and quantization errors
- Deterministic results (important if you have a simulator to verify your control equations)

There is cost in significant design complexity for you, the designer. But you won't be using large sheets of paper. You will be using tools we never dreamed of that are available today that will do most of the heavy lifting.

My next article will be my last for Embedded in Thin Slices. I will attempt to distill my 50 years of embedded system design—but of course, only in one thin slice!

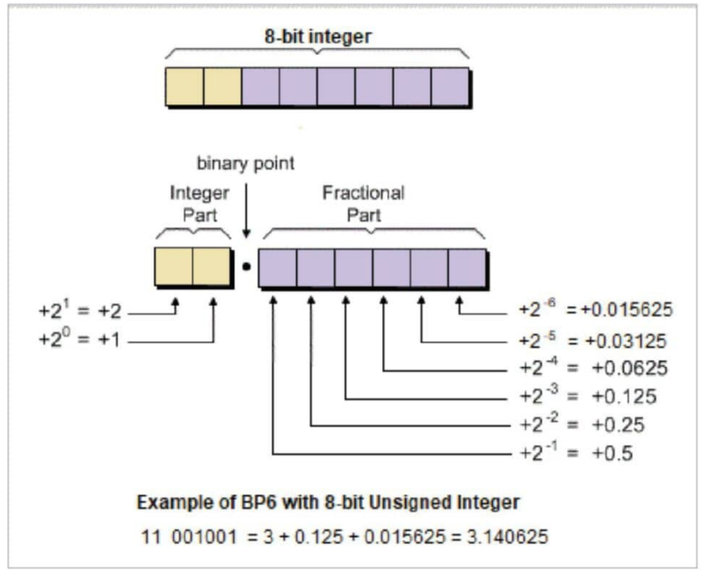


FIGURE 3
Pi as BP6.

ABOUT THE AUTHOR

Bob Japenga has designed embedded systems since 1973. From 1988-2020, Bob led MicroTools, an engineering firm specializing in creating a variety of real-time embedded systems across a wide range of industries. Bob has been awarded 11 patents in many areas of embedded systems and motion control. Now retired, he enjoys building electronic projects with his grandchildren. You can reach him at Bob@ListeningToGod.org.

COLUMNS

ONEKEY's New Automated Cybersecurity Testing for Embedded Software Available in Compliance with RED II EN18031 Standard

As of August 2025, connected digital products will be required to comply with the EU Directive—Radio Equipment Directive (RED) II EN 18031. In response, Düsseldorf-based cybersecurity company ONEKEY has integrated testing for RED II, EN 18031-1 into its automated cybersecurity platform.

"We are expanding the capabilities of our automated platform to include this key EU directive, which is mandatory for all manufacturers of products containing radio modules. This underlines our commitment to providing customers with comprehensive and up-to-date compliance support," said Jan Wendenburg, CEO of ONEKEY. "The new automated RED II testing is primarily aimed at manufacturers and OEMs of smart devices, machines, and systems. In addition, the ONEKEY platform's Compliance Wizard already includes other relevant standards such as the Cyber Resilience Act, IEC 62443-4-2, ISO 303645, UNR 155, UK PSTI, and many more. Coverage of these standards is being continuously expanding."

Automated RED II Analysis and Structured Documentation via Decision Tree—Key application areas include fully automated technical software analysis to identify vulnerabilities and compliance gaps, as well as structured documentation using a decision tree. This enables customers to automatically check specific technical RED II requirements within minutes through binary analysis. Non-compliant areas—the gaps between the standard and the software—are automatically identified and documented with supporting technical evidence.

The remaining RED II EN18031-1 requirements can then be addressed step by step through an intelligent decision tree, directly linked to the specific RED provisions. In many cases, answers are pre-filled automatically based on previous analysis results, while manual review and adjustment remain possible at any time.

Background on RED II EN 18031—The Radio Equipment Directive (RED) 2014/53/EU of the European Union is a key regulatory framework for connected devices, systems, and machines, as it defines essential requirements for radio equipment sold in the EU market. With the growing number of industrial machines, sensors, actuators, and other digital products being connected via wireless communication (Internet of Things, IoT, Industry 4.0), these products now fall under the scope of the updated RED II EN 18031, or Delegated Regulation (EU) 2022/30, which was introduced in 2022.

The directive is intended to ensure that these devices provide electromagnetic compatibility and appropriate cybersecurity to prevent interference in radio communications. It requires manufacturers to ensure that any product using radio technologies complies with the requirements of the directive before being placed on the European market. This applies not only to traditional communication devices but increasingly to industrial products that integrate radio modules for use in smart factories, logistics, or the control of autonomous machinery.

These requirements for cybersecurity and electromagnetic compatibility are particularly critical in industrial environments, where malfunctions or cyberattacks on connected systems could have far-reaching consequences.

ONEKEY | onekey.com

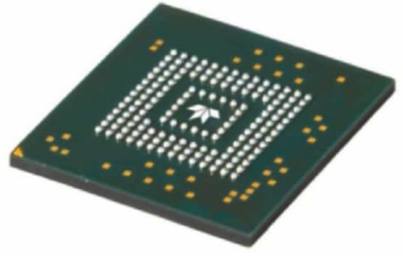
Teledyne HiRel Semiconductors Launches eMMC 5.1 Module

Teledyne HiRel Semiconductors is pleased to announce the release of its industrial-grade embedded MultiMediaCard (eMMC) module. Featuring 128GB of eMMC 5.1-compliant storage in a compact 153-ball FBGA package, this device is engineered to deliver reliable, high-speed performance in harsh environments.

Qualified over the industrial temperature range of –40°C to +85°C, this eMMC module is well-suited for embedded applications in aerospace, defense, industrial automation, and ruggedized edge computing. Its integrated controller handles wear leveling, bad block management, and error correction (ECC), ensuring long-term data integrity and simplified system integration.

"With eMMC continuing to serve as a foundational technology for embedded systems, we're proud to offer a drop-in solution with long-term supply assurance and robust performance," said Mont Taylor, Vice President of Business Development, at Teledyne HiRel. "This release supports our customers' demand for reliable, off-the-shelf storage solutions that integrates easily into space-constrained, mission-critical systems."

Available now, the module supports JEDEC-standard eMMC 5.1 commands, operates from a single 3.3V supply (I/O at 1.8V), and delivers sustained read/write speeds suitable for OS boot, data logging, and code storage. Its long-term availability helps ensure continuity and support across multi-year program lifecycles.



Teledyne HiRel Semiconductors
teledyne-ade.com

TDK Expands Ultra-Low Voltage TVS Diode Portfolio for High-Speed Consumer Interfaces

TDK Corporation has expanded its ultra-low clamping voltage TVS diode lineup with three new models in the SD0201 series, tailored for high-speed consumer electronics interfaces. These TVS diodes offer compact protection for USB Type-C, HDMI, DisplayPort, and Thunderbolt connections in smartphones, laptops, tablets, wearables, and networking devices.

Each component comes in a 0201 chip-scale package (CSP), measuring just 0.58 x 0.28 x 0.15 mm (L x W x H) for space-constrained designs. Having working voltages of ±1 V, ±2 V, and ±3.6 V, the new TVS diodes safeguard sensitive circuits from electrostatic discharge (ESD) and transient surges. This means they meet IEC 61000-4-2 standards with ESD discharge robustness up to ±15 kV and support surge current handling up to 7 A, depending on the variant. Their symmetrical design enables bidirectional protection and thus flexible PCB routing, while very low leakage currents and dynamic resistance as low as 0.16 Ω enhance application-level efficiency.

The individual components differ in their DC working voltage and parasitic capacitance: the SD0201SL-S1-ULC101 (ordering code B74121U1036M060) has a working voltage of ±3.6 V and a parasitic capacitance of 0.65 pF, the SD0201-S2-ULC105 (B74121U1020M060) of ±2 V and 0.7 pF, and the SD0201SL-S1-ULC104 (B74121U2010M060) of ±1 V and 0.15 pF.

TDK's new ultra-low clamping voltage TVS diodes are engineered to meet the challenges in today and tomorrow's consumer electronics, ensuring reliable performance and safeguarding sensitive electronic devices from damaging transients.

Main applications include high-speed interfaces in smartphones, laptops, tablets, wearables, network communication devices, and other portable devices with tight space requirements

Main features and benefits:

- Bidirectional ESD protection of one high-speed I/O line
- ESD discharge robustness up to 15 kV
- Very low clamping voltage
- Low capacitance
- Low leakage current
- Chip-scale package 0201 (0.58 x 0.28 mm) with a height of 0.15 mm

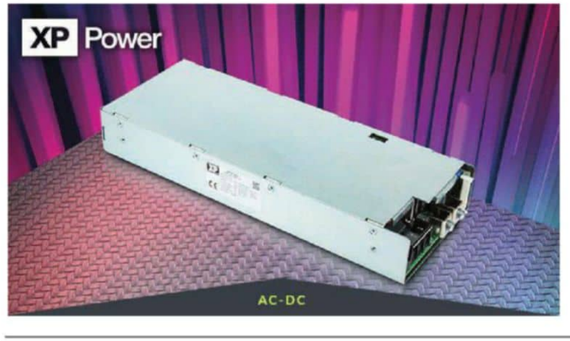
TDK Corp. | www.tdk.com

XP Power Introduces Digital Programmable, Compact 1.5kW AC-DC Power Supplies

NXP Power has introduced the HPA1K5 Series. The HPA series of programmable, scalable, configurable AC-DC power supplies deliver 1.5kW of power from a universal single phase input with single outputs ranging from 24 to 60VDC.

The HPA1K5 series offers industry leading power density in a compact 1U package, high efficiency, industrial and medical safety approvals, class B conducted emissions, 2 x MOPP isolation and Semi F47 compliance, making it suitable for a wide range of medical, semiconductor, industrial and analytical instrumentation applications. Flexible, scalable, programmable low profile 1.5kW AC-DC power supplies adds two new voltage outputs, 36VDC & 60VDC, to its existing range of 24VDC & 48VDC outputs.

HPA1K5's robust digital architecture, which includes digital controls and monitors, constant current/constant voltage operation, slew rate control and fault detection, allows designers to seamlessly integrate the product into their systems. Its Graphic User Interface enables user-defined controls, signals, and alarms to tailor the power solution to the requirements of the end application.



HPA1K5 Features Include:

- Nominal output voltages 24/36/48/60VDC
- Programmable constant voltage (0-105%) & constant current (0-110%)
- Parallel operation, scalable up to 7.5kW
- Multiple digital protocols: I²C PMBus; CANopen; MODBUS; SCPI
- Graphical User Interface (GUI)
- ITE and Medical safety approval

XP Power | xppower.com

Infineon and Delta Collaborate on High-Density Power Modules to Accelerate Data Center Power Architecture

Infineon Technologies AG announced the strengthening of its existing collaboration with Delta Electronics, Inc., in the development of state-of-the-art high-density power modules capable of enabling vertical power delivery to AI processors in hyperscale data centers. Together, the companies are taking a next step to drive decarbonization and digitalization in AI data centers further.

The partnership leverages Infineon's ultra-thin silicon (MOSFET) chip technology and embedded packaging expertise, as well as Delta's industry-leading power module design and manufacturing capabilities. This results in highly dense modules with exceptional efficiency to enable vertical power delivery (VPD) architecture to the xPUs in hyperscale data centers. The usage of vertical power delivery modules in comparison to a lateral mounted discrete solution can save up to 150 tons of CO_2 over an expected lifetime of three years per rack. Assuming that future hyperscale data centers will consists of up to 100 server racks, the amount of carbon dioxide saved is equivalent to the CO_2 emissions of 4000 households per year.

Infineon's OptiMOS silicon-based 90A integrated power-stage solution is being utilized by Delta to develop VPD modules. The use of vertical power delivery is a key factor in improving system efficiency as it allows for a more direct and compact power delivery path. By delivering power vertically, rather than horizontally, the VPD modules reduce power delivery network losses in the system. This, in turn, enables the modules to achieve higher power density and efficiency, while also reducing the amount of heat generated due to less power loss. Additionally, the vertical power delivery design also frees up space on the system board, allowing hyperscalers a more efficient use of space and the development of more compact and dense data center designs reducing total-cost of ownership.

As the AI era continues to advance, the demand for high-power data centers is increasing exponentially. To meet this demand, data centers require power delivery solutions that can support the growing power needs of AI training and inferencing applications. Furthermore, the increasing densification of IT racks, driven by the integration of tightly interconnected xPUs requires the development of higher voltage power distribution solutions that can efficiently deliver power to these high-density systems. In response to these challenges, Infineon and Delta are cooperating in the development of next-generation power delivery solutions for AI data centers that will require 1MW per rack in the near future.

Alongside the company's broad portfolio in gallium nitride (GaN) and silicon carbide (SiC) power solutions, Infineon is the global leader in silicon (Si) MOSFETs that are designed to enable best-in-class performance, reliability and ease-of-use in power switching, control and conversion. This means more efficiency, power density and cost effectiveness for products and system solutions.

Infineon Technologies | infineon.com

Advanced Energy Launches 1300W and 1600W Ultra-Efficient DC-DC Converter

Advanced Energy Industries, Inc., a global leader in highly engineered, precision power conversion, measurement and control solutions, announced expansion of its ultra-efficient non-isolated bus converter (NIBC) family with two new quarter brick modules for 48V power conversion in advanced information and communication technology (ICT) equipment including Artificial Intelligence (AI) server, compute and networking and in industrial applications such as robotics and test and measurement.

The new Advanced Energy Artesyn NDQ1300 1300 W and NDQ1600 1600 W quarter-brick modules operate with peak efficiencies up to 98%, making them ideal for high-performance applications where power density is a critical design criterion. Each of the modules can convert a 48V input into a fully-regulated 12V output for non-isolated, low-voltage, high-current power stages as well as PCIE slots and memory devices.

"Today's power system designers are focusing on efficiency and form factor as they look to deliver the low loss power conversion essential to emerging AI and next-generation industrial applications," said Joe Voyles, Advanced Energy's vice president, Industrial Power Conversion, System Power. "The NDQ1300 and NDQ1600 deliver high power density and high efficiency across the range, offer a standard footprint to optimize performance, reduce power use and size, and simplify design."

The new NDQ products feature a flat efficiency curve that ensures modules deliver optimum power conversion across a wide load range. An integrated PMBus interface supports flexible digital control and monitoring. Current-share and remote sensing options enable multiple power supplies to connect in parallel, supporting higher load current or redundancy. In addition, the NDQ products use an inherently safe transformer-based topology that is resilient to transient loads and makes designing applications for inrush current-control on start-up easier.

The new NDQ modules leverage an advanced baseplate for better thermal management and heatsink integration. Standard sizes make power scaling straightforward, while the compact design protects downstream circuits.

Advanced Energy Industries, Inc.
advancedenergy.com

Renesas Introduces 64-bit RZ/G3E MPU for High-Performance HMI Systems Requiring AI Acceleration and Edge Computing

Renesas Electronics Corp., a premier supplier of advanced semiconductor solutions, announced the launch of its new 64-bit RZ/G3E microprocessor (MPU), a general-purpose device optimized for high-performance Human Machine Interface (HMI) applications. Combining a quad-core ARM Cortex-A55 running at up to 1.8GHz with a Neural Processing Unit (NPU), the RZ/G3E brings high-performance edge computing with AI inference for faster, more efficient local processing. With Full HD graphics support and high-speed connectivity, the MPU targets HMI systems for industrial and consumer segments including factory equipment, medical monitors, retail terminals, and building automation.

High-Performance Edge Computing and HMI Capabilities—At the heart of the RZ/G3E is a quad-core ARM Cortex-A55, a Cortex-M33 core, and the Ethos-U55 NPU for AI tasks. This architecture efficiently runs AI applications such as image classification, object recognition, voice recognition and anomaly detection while minimizing CPU load. Designed for HMI applications, it delivers smooth Full HD (1920×1080) video at 60fps on two independent displays, with output interfaces including LVDS (dual-link), MIPI-DSI, and parallel RGB. A MIPI-CSI camera interface is also available for video input and sensing applications.

The RZ/G3E is equipped with a range of high-speed communication interfaces essential for edge devices. These include PCI Express 3.0 (two lanes) for up to 8Gbps, USB 3.2 Gen2 for fast 10Gbps data transfer, and dual-channel Gigabit Ethernet for seamless connectivity with cloud services, storage, and 5G modules.

Low-Power Standby with Fast Linux Resume—Starting with the third-generation RZ/G3S, the RZ/G series includes advanced power management features to significantly reduce standby power. The RZ/G3E maintains sub-CPU operation and peripheral functions while achieving low power consumption around 50mW and around 1mW in deep standby mode. It supports DDR self-refresh mode to retain memory data, enabling quick wake-up from deep standby for running Linux applications.

Renesas | renesas.com

Dual-Core MCU Support Added in Latest NECTO Studio IDE from MIKROE

MIKROE, the embedded solutions company that dramatically cuts development time by providing innovative hardware and software products based on proven standards, has announced that version 7.3.0 of its NECTO Studio Integrated Development Environment (IDE) now supports dual-core MCUs, allowing designers to program and debug each core independently while flashing merged firmware in the background. This enables tasks to be run in parallel, increases power efficiency, and facilitates the creation of complex systems such as real-time control, user interfaces, and AI-based edge devices. Also new in the latest version, NECTO Studio's Productivity Hub combines powerful AI and automation tools to save time and increase productivity. The Visual Prompt Generator speeds code generation, and custom hardware configurations can be easily managed, reused, and shared using the integrated Custom Board Tool.

Comments Nebojsa Matic, CEO of MIKROE: "Our goal is to deliver the world's first multi-architecture, multi-language IDE with the best user experience. We are excited to add dual-core support and provide easy, integrated access to AI tools from within NECTO Studio IDE."

Dual-core support within NECTO Studio v7.3.0 brings: independent Start / Stop / Pause / Restart / Step Over / Step In / Step Out for each core; core-specific disassembly view; run to breakpoints; simultaneous flashing with automatic HEX merging; and Integrated project explorer support for dual-core projects. STM32 dual-core MCUs are currently supported—this will shortly be extended to other vendors and architectures including NXP, Renesas and more in future updates.

The Productivity Hub, included in NECTO Studio for the first time, enables embedded developers to build advanced embedded applications with multiple Click boards in minutes, guided by a simple and intuitive three-step wizard. First, NECTO automatically detects and preselects the setup from an active project. Second, quickly browse and select from more than 1800 Click boards using the searchable list which can be filtered by category, function, or communication protocol. Third, if required, select a display. Upon completion of the wizard process, NECTO Studio automatically creates a tailored AI prompt for the NECTO Studio AI Code Assistant, which generates working code based on the selections made. Simply copy, paste, build, and run.

A redesigned Custom Board Tool is also fully integrated into the Productivity Hub, providing an easier and more effective way to manage custom hardware configurations. It enables designers to visually assign mikroBUS pin connections, use a custom board in any local project, and to share or publish a custom board for team-collaboration and reuse.

MIKROE | mikroe.com

NECTO STUDIO
IDEAL CODING
UPDATE 7.3.0
WE ARE ON THE MISSION: BUILDING WORLD'S FIRST
MULTI-ARCHITECTURE, MULTI-LANGUAGE IDE WITH THE BEST UX

PRODUCT NEWS

New High-Voltage Programmable Resistor Modules from Pickering Provide a Simple Solution for Applications Requiring Voltage Handling up to 1.2kV

Pickering Interfaces, the leading supplier of modular signal switching and simulation solutions for use in electronic test and verification, has announced a new family of high voltage programmable resistor modules in a compact single-slot PXI and PXIe form factor, model 40-230 (PXI) and 42-230 (PXIe), providing a simple solution for applications requiring voltage handling up to 1.2kV. Part of Pickering's expanding range of programmable resistor modules, the new module is available in 70 standard builds, including four single-channel builds that offer a wide variety of resistance ranges and resolution capabilities, playing a pivotal role in simulating real-world resistive loads and fault conditions for high-voltage devices-under-test (DUTs) across automotive, aerospace, and energy sectors.

The 40/42-230 high-voltage programmable resistor module plays a critical role in simulating real-world resistive loads and fault conditions for high-voltage DUTs such as electric vehicle (EV) power inverters, battery management systems (BMSs), and some aerospace components. Engineers can utilize these modules for critical tasks, including voltage isolation check testing, which ensures the safe functionality of high-voltage systems and compliance with safety standards such as ISO 6469 and IEC 60664.

The module allows engineers to emulate resistance sensor values, such as those from thermistors and RTDs, in high-voltage environments, enabling safe testing without needing the actual sensor hardware. For added test coverage, when used in the hardware-in-the-loop (HIL) simulation testing of electronic control units (ECUs), each channel of the 40/42-230 can be set to short and open circuit conditions to verify the controller's response to a faulty sensor or connection.

With up to four channels available from a single slot, the module offers multi-channel flexibility and a wide resistance range—from 1Ω to 76.8MΩ—in a compact single-slot PXI or PXIe form factor, as well as resistance resolution down to 0.125Ω. The module's hardware interlock and simulation features ensure safety and facilitate diagnostics. Key applications range from high-voltage test environments, including automotive EV and aerospace systems, to applications requiring rapid reconfiguration of resistance, or test systems with limited PXI slots (requiring multichannel modules)—as well as HIL simulation platforms needing fault insertion.

Software control is simplified by using resistor value calls. The module calculates the resistance setting closest to the requested value and sets that value. The user can then interrogate the module to find the actual setting used. To help ensure long-term accuracy, an optional cable assembly may be attached to the module in place of the DUT, enabling simple calibration using a digital multimeter (DMM) to verify the resistor channels.

The 40/42-230 high-voltage modules are fitted with a hardware interlock, which provides greater safety than typical software protection. When activated, the interlock will return all relays to their default unpowered state (assuming the switches are fully functional) and provide error notification. The interlock feature can be daisy-chained between additional hardware interlock-enabled modules, allowing one signal to disable multiple cards.

Pickering Interfaces | pickeringtest.com

Concurrent Launches Bragi: A High-Performance GPGPU Card Built with NVIDIA Blackwell Architecture

Concurrent, a leader in embedded computing solutions, announced the release of Bragi, a rugged 3U OpenVPX GPGPU processing card in partnership with EIZO Rugged Solutions. Bragi combines the latest NVIDIA RTX PRO 4000 and 5000 GPUs, based on the new Blackwell architecture, with the proven reliability and integration support that customers expect from Concurrent.

Bragi is designed for demanding C5ISR and HPEC applications requiring high-performance AI processing at the edge. It is fully SOSA aligned and supports both conduction-cooled and air-flow-through variants, making it suitable for deployment in space-constrained and thermally challenging environments.

Miles Adcock, CEO, said: "We've developed our partnership with EIZO Rugged Solutions over many years, and are delighted that we have the rare ability to sell their flagship AI enabling product as one of our own. In addition to enhancing our portfolio, this is highly reassuring to customers who wish to be confident that the elements of a system will work flawlessly together. We've invested in the technical ability to deeply integrate and support the GPGPU's from EIZO Rugged Solutions as part of our ongoing mission to provide customers with the very latest technology."

Bragi's design addresses the requirements of modern defence and aerospace platforms, particularly those balancing advanced AI workloads with SWaP constraints and legacy system compatibility. Built around Blackwell architecture, the product delivers up to 1,824 FP4 AI TOPS, enhanced video encoding/decoding, and high-bandwidth memory access via PCIe Gen 5 and GDDR7 ECC memory.

A Trusted Collaboration—Concurrent and EIZO Rugged Solutions have a proven track record of working together to deliver high-performance embedded computing products. Bragi builds on this foundation, enabling fast deployment for customers needing reliable, high-throughput AI and video processing in the field.

Concurrent | concurrent.tech

NEW PRODUCT SUBMISSIONS— E-mail: product-editor@circuitcellar.com

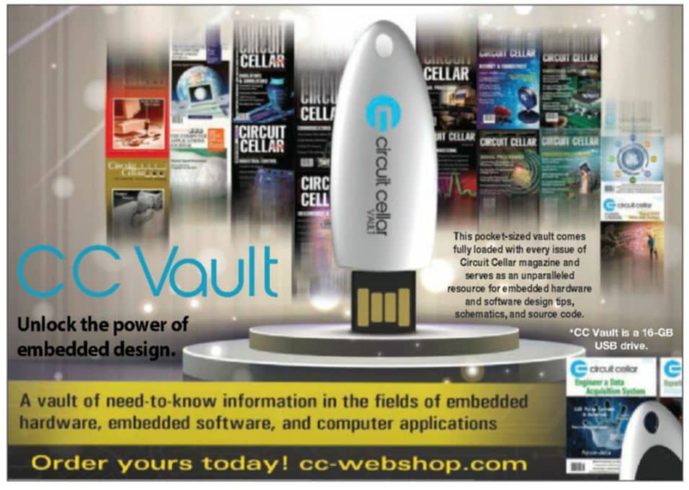
IDEA BOX

The Directory of PRODUCTS & SERVICES

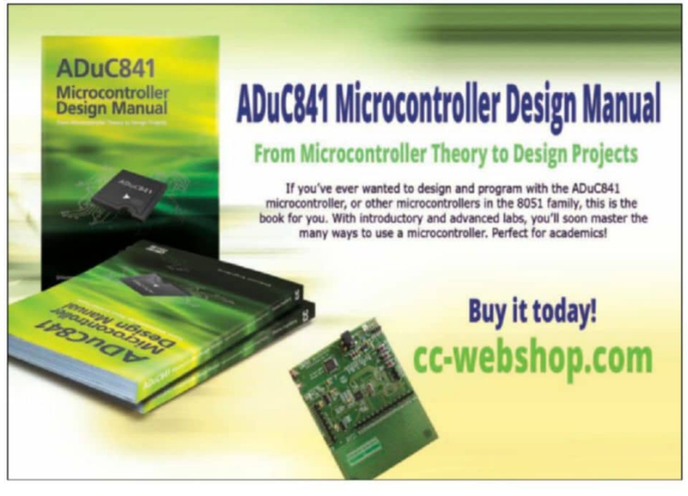
AD FORMAT:
Advertisers must furnish digital files that meet our specifications (www.circuitcellar.com/mediakit).

All text and other elements MUST fit within a 2" x 3" format.
E-mail adcopy@circuitcellar.com with your file.

For current rates, deadlines, and more information contact Hugh Heinsohn at 757-525-3677 or Hugh@circuitcellar.com


CC Vault
Unlock the power of embedded design.
This pocket-sized vault comes fully loaded with every issue of Circuit Cellar magazine and serves as an unparalleled resource for embedded hardware and software design tips, schematics, and source code.
*CC Vault is a 16-GB USB drive.
A vault of need-to-know information in the fields of embedded hardware, embedded software, and computer applications
Order yours today! cc-webshop.com

When it comes to robotics, the future is now!
Advanced Control Robotics simplifies the theory and best practices of advanced robot technologies, making it ideal reading for beginners and experts alike.
With this book, you'll learn about:
• Communication
• Technologies
• Control Robotics
• Embedded Technology
• Programming Language
• Visual Debugging... and more
ADVANCED CONTROL ROBOTICS
Get it today, cc-webshop.com


ADuC841 Microcontroller Design Manual
From Microcontroller Theory to Design Projects
If you've ever wanted to design and program with the ADuC841 microcontroller, or other microcontrollers in the 8051 family, this is the book for you. With introductory and advanced labs, you'll soon master the many ways to use a microcontroller. Perfect for academics!
Buy it today!
cc-webshop.com

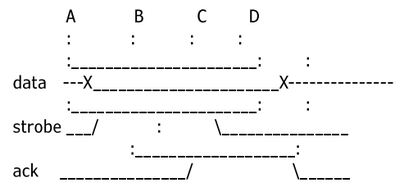
embeddedTS
embeddedTS.com
TS-7100-Z
Powered by the high performance, ultra-efficient Arm® Cortex®-A7 based i.MX6UL CPU
...In an enclosure as small as two decks of cards
(2.4" x 3.6" x 1.7")
Best In Show
Embedded Computer Design
Made in the USA
with global parts

TEST YOUR EQ

Contributed by David Tweed

Answer 1 —

In the two-wire, four-edge handshake, one wire is driven by the transmitting device; we'll call it "strobe". The other wire is driven by the receiving device; we'll call it "acknowledge", or just "ack".



The transmitting device initiates a transfer by driving valid data on the data wires and asserting strobe (event A). When the receiving device sees strobe asserted, it reads the data. It delays long enough to insure that its own setup and hold time requirements are met and then asserts ack (event B).

When the transmitting device sees ack asserted, it can release the bus, and negate strobe (event C). And when the receiving device sees strobe negated, it negates ack (event D) and puts the bus back into the idle state, ready for the next transfer.

Answer 2 —

Early microprocessor-based systems used little more than bit-banging to implement communication protocols. At the lowest level, this means that the softwrae could sample the level of an input wire periodically, and it could update the state of an output wire.

Note that events A and C are signaled by the transmitting device, and events B and D are signaled by the receiving device, and these events occur in strict alternation, insuring that both devices remain in sync with each other.

Neither device needs to remember the previous state of a wire; it just needs to know where it is in the protocol and what level to look for next.

Answer 3 —

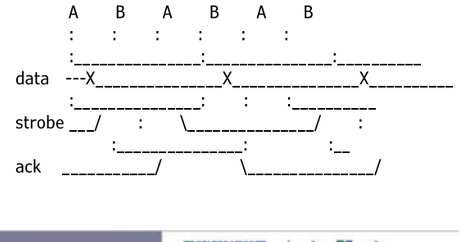
Multiple receiving devices could be supported by defining some of the data wires to be address wires. When the strobe occurs, only the receiving device that is addressed would respond.

Note that the bus is "busy" between events A and D — i.e., whenever either strobe or ack is asserted. But when they're both negated, the bus is idle. This fact could be used to allow switching among multiple transmitting devices. However, arbitrating among them for access to the bus would require additional logic.

Answer 4 —

As mentioned before, doing edge detection on a signal requires each device to keep track of the previous state of the signal. But noting that events C and D are used only to get the bus back into its idle state, using edge detection would eliminate the need for them altogether.

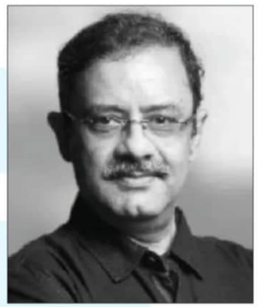
Instead, the act of toggling the state of strobe would be enough to signal the receiving device that data is ready, and the act of toggling the state of ack would serve to complete the transfer. The next transfer could be initiated right away by the next toggle of strobe.



The Future of Microsegmentation Cyber Defense

What Is OT Breach-Ready Cyber Defense?

Microsegmentation Is Strategic Defense Against Intrusion Damage



By
Agnidipta Sarkar,
Chief Evangelist,
ColorTokens

Breaches happen. When microsegmentation is implemented properly, it's a strategy that can limit the damage a breach can do, keeping an intruder limited to a small part of an OT or IoT network. Understanding microsegmentation is now an important part of an embedded systems education for engineers and developers.

Breach-ready cyber defense is a proactive and resilient approach designed to ensure that operational technology (OT) environments—such as manufacturing lines, refineries, pipelines, rail control systems, or telecom switching networks—can continue operating even if a cyber breach occurs. This approach goes beyond preventive cybersecurity controls and best practices that focus on making it difficult for attackers to gain access, prioritizing active containment upon early detection to prevent attacks from expanding beyond the initial point of entry.

Unlike traditional perimeter-focused security, such as that shown in **Figure 1**, which assumes you can keep attackers out, breach-ready cyber defense assumes compromise is inevitable and focuses its activities on:

- Prevention: Anticipating attacks through attack path mapping, surface reduction, interception modeling, and lateral movement control
- Response: Containing attacks mid-stream by quarantining zones, isolating unaffected systems, and disrupting breaches
- Resilience: Evolving organizational endurance through effective containment, recovery, and failover capabilities

In OT environments, this is more than just avoiding data theft—it's about preventing loss of life, environmental disasters, or crippling national infrastructure and ensuring reliable efficiency. OT breach-ready cyber defense is the bedrock of society in general and absolutely paramount to the growth of the nation.

URGENT FOR CRITICAL INFRASTRUCTURE

Let us look at reality. Despite increasing investments into cybersecurity, a market estimated to grow at a rapid pace from $276.63 billion in 2025, cyberattacks are not slowing down.

Increased digitalization and adoption of Industry 4.0 and 5.0 are resulting in the explosion of the global digital attack surface:

- Manufacturing: Smart factories integrate robots, sensors, MES, ERP, and supplier networks
- Pharma: Batch control systems tied to cloud analytics and regulatory portals
- Oil & Gas: Offshore rigs and refineries with hybrid IT/OT/IoT ecosystems
- Transportation: Signaling, safety, and scheduling systems connected to corporate IT
- Telecom: Core network elements linked to orchestration platforms and customer systems

This is not all. Digital industrial systems are fundamentally different from traditional IT, making it challenging to apply traditional

FIGURE 1
In a traditional network, breaking through the perimeter into the network core allows access to almost every node on the network without additional security checks. (Image by jeferrb from Pixabay)

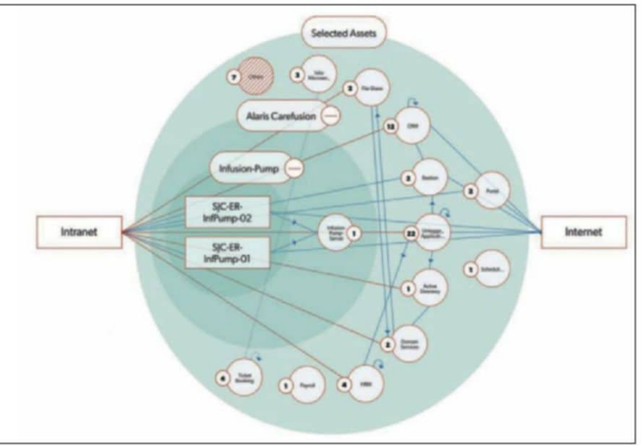


FIGURE 2
Color-coded lines indicating in-policy and out-of-policy traffic. Views can be organized by up to 20 dimensions. Users can see different views for their role.

cybersecurity solutions. The complexities include legacy OT devices with no built-in security, flat networks where one compromised device can take down an entire plant, IT–OT convergence blurring trust boundaries, ransomware and nation-state attacks targeting physical disruption (not just data), and regulatory pressure (e.g., NIS2, CISA advisories, NERC CIP, ISA/IEC 62443).

Industrial organizations must invest in OT breach-ready cyber defense today. Delays can cause safety system failure, production halts, environmental damage, cascading effects on national supply chains, and even loss of life—think Colonial Pipeline, Triton/TRISIS safety controller malware, attacks on various Hospital medical systems, or NotPetya's global manufacturing shutdowns.

MICROSEGMENTATION IS FOUNDATIONAL

Microsegmentation is the practice of breaking the OT network into small, controlled security zones or micro-perimeters with tightly enforced communication rules on the conduits that connect them, which can be disconnected as soon as a cyberattack attempt is detected. Instead of one "big flat" OT network, you have multiple isolated zones of excellence that can only talk in pre-approved, minimal ways, in a topology illustrated in Figure 2. And this means that even if an attacker manages to beat the traditional cybersecurity controls, the attack remains confined to that specific micro perimeter.

When deployed for breach ready cyber defense, microsegmentation can:

- Isolate safety systems from non-safety systems (TRICONEX or SIS controllers in separate, locked-down zones)
- Segment critical ICS components so that a compromise in a monitoring system doesn't jump to control logic
- Control lateral movement, ensuring attackers can't move from one zone to another or reach critical assets
- Protect cyber-physical systems by ring-fencing PLCs, RTUs, and SCADA servers from unnecessary access
- Enforce least privilege — only the exact devices, protocols, and ports needed for the function are allowed
- Contain breaches at the edge where IT, cloud, and industrial systems meet

There are other advantages of using microsegmentation, including a highly reduced attack surface, thus the security alerts reaching an OT cyber defense center (also called OT SOC) would become more real; a highly reduced blast radius of each digital system, thus attempts of lateral movement would become malicious much faster; and a faster mean time to respond to cyberattacks from days or hours to minutes.

MICROSEGMENTATION IS RESILIENCE BEDROCK

Without microsegmentation, the consequences of a breach can be severe. One compromised laptop can pivot across the plant network. A single compromised VPN credential can open the door to human harm or ransomware or malware in quality control monitoring could jump into process control logic. Attackers can move from IT to OT via shared services (active directory, file servers, historian databases, etc.)

In the past, we've seen examples of attacks on OT systems including:

- Attackers attempted to poison the water supply by increasing the level of sodium hydroxide (lye) in a Florida water treatment plant in 2021.
- Attackers disrupted a pressure regulation pump and defaced a PLC-HMI system in 2023.
- A ransomware attack forced Tower Semiconductor to halt operations at some manufacturing facilities.

With microsegmentation, the attack surface shrinks dramatically, reducing threat intel overload. By blocking the lateral movement of potential breaches, the subsequent impact is limited to one zone and one micro perimeter, speeding up incident response time because compromise is localized. As a result, building cyber defense becomes scalable; once segmentation policies exist, they can be rolled out across plants, sites, and geographies. With emerging capabilities to integrate with existing EDR, retrofitting cybersecurity procedures to establish a "breach-ready" cyber defense strategy for OT systems becomes a one-day activity.

Today, OT breach-ready cyber defense is not optional—it is mission-critical for every industrial organization, ensuring human and societal safety and allowing them to continue operating critical parts of their supply chain. This allows them to operate in a regulatory-heavy environment and afford days or weeks of acceptable downtime in small areas.

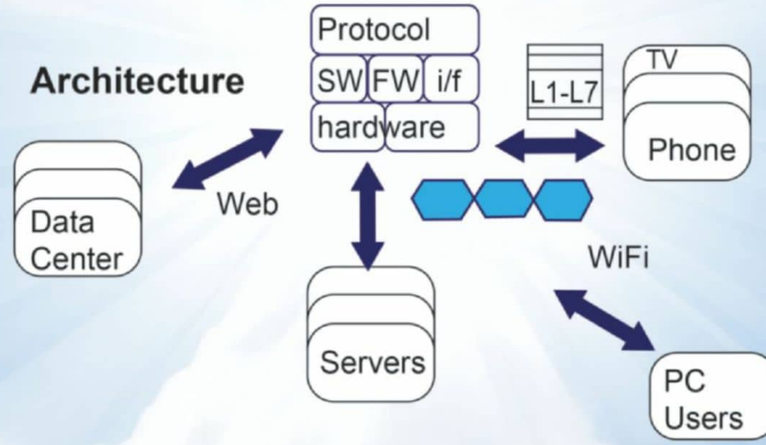
Microsegmentation serves as the first step of OT breach readiness. By creating a structure where cyberattacks can be immediately contained, unaffected areas can reliably continue normal operations while the effects of the breach are managed quicker and with higher efficacy. Without it, every other cyber control is weakened by an attacker's ability to roam freely.

RESOURCES
ColorTokens
www.colortokens.com

ABOUT THE AUTHOR

Agnidipta Sarkar is a leading cybersecurity evangelist who has been an Auditor, a Consultant, a Practice Leader, a Risk Officer, and a CISO. Agni leads the ColorTokens CISO Advisory program, arming CISOs, CXOs, and Boards to build cyber defense and digital resilience and adopt zero trust. Agni is a contributing member of ISO, Cloud Security Alliance, NIST, ISA and BCI. He is leading a panel to publish the first digital resilience standard for the Bureau of Indian Standards. Before ColorTokens, Agni was the Group CISO at a major pharma organization.

SeaFire® Micros, Inc.
IP & Technology for Cloud Networks



Design through Production
Military — Commercial

Hardware (Analog/Digital)
Firmware • Software • FPGA

Patented Network Technology | Unique Cloud Technology
No Compete Government Procurement
No Compete Government Funding

20% DISCOUNT on 1st Purchase

info@seafire.com | seafire.com

TS-9370

Low Power, High Performance SBC

The TS-9370 is a versatile and power-efficient industrial SBC built for rugged applications. Powered by the NXP i.MX9352 64-bit ARM® architecture plus two Cortex®-A53 processors operating up to 1.7 GHz, it delivers robust performance with low power consumption.

32 GB eMMC (upgradable)

2 GByte LPDDR4 RAM

MikroBUS socket
For MikroE Click Boards

Made in the USA
with global parts

ISA SUMMIT & EXPO
OCT 5-7
LAKE BUENA VISTA, FL

ITC TELEMETRY
OCT 20-23
LAS VEGAS, NV

EMBEDDED WORLD
NOV 4-6
ANAHEIM, CA

embeddedTS

TS-4100

Ultra-Efficient and Fully Featured

The TS-4100 features an NXP i.MX 6UltraLite CPU, industrial temp range, FPGA, eMMC, and customizable I/O. Comes preloaded with Debian Stretch.

TS-7100-Z

As Small as Two Decks of Cards

Powered by an Arm® Cortex®-A7 i.MX6ul CPU, it runs efficiently at <2W, featuring Dual Ethernet, Bluetooth, WiFi, and NimbeLink/Digi sockets.

www.embeddedTS.com